



AFRL-RH-AZ-TR-2008-0020

Using LSA to Compute Work Sense Frequencies

**Esther Levin
Mehrbod Sharifi**

**Research Foundation of CUNY/City College
230 W. 41st Street, 7th Floor
New York, NY 1036**

**February 2008
Final Report for July 2005 to November 2007**

**DESTRUCTION NOTICE – Destroy by any method
that will prevent disclosure of contents or
reconstruction of this document.**

Distribution is Unlimited.

**Air Force Research Laboratory
Human Effectiveness Directorate
Warfighter Readiness Research Division**

NOTICES

This report is published in the interest of scientific and technical information exchange and its publication does not constitute the Government's approval or disapproval of its idea or findings.

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

Qualified requestors may obtain copies of this report from the Defense Technical Information Center (DTIC) at <http://www.dtic.mil>.

AFRL-RH-AZ-TR-2008-0020 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//signed//
JERRY T. BALL
Lab Contract Monitor

//signed//
HERBERT H. BELL
Technical Advisor

//signed//
DANIEL R. WALKER, Colonel, USAF
Chief, Warfighter Readiness Research Division
Air Force Research Laboratory

REPORT DOCUMENTATION PAGEForm Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 22022008		2. REPORT TYPE Final Technical Report		3. DATES COVERED (From - To) Jul 2005 to Nov 2007	
4. TITLE AND SUBTITLE Using LSA to Compute Word Sense Frequencies				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER FA8650-05-1-6637	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Esther Levin, Mehrbod Sharifi				5d. PROJECT NUMBER	
				5e. TASK NUMBER HA	
				5f. WORK UNIT NUMBER 1123AS51	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Research Foundation of CUNY/City College 230 W. 41 st Street, 7 th Floor New York, NY 1036				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory Human Effectiveness Directorate Warfighter Training Research Division 6030 South Kent Street Mesa AZ 85212-6061				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL; AFRL/RHA	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-RH-AZ-TR-2008 - 0020	
12. DISTRIBUTION / AVAILABILITY STATEMENT Distribution A. Approved for public release; distribution unlimited. (Approval given by 88 ABW/PA. WPAFB 08-3345)					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT This document describes a project to explore the use of Latent Semantic Analysis (LSA) and statistical clustering techniques for automatically identifying word senses and for estimating word sense frequencies from application relevant corpora. The hypothesis is that LSA can be used to compute context vectors for ambiguous words that can be clustered together – with each cluster corresponding to a different sense of the word. The document is organized as follows: the first section includes a short introduction to LSA, an introduction to the context-group discrimination paradigm adopted in the project, and a description of the corpus used in the experiments. Section 2 describes the investigation of the effect of LSA dimensionality on sense discrimination accuracy. Overall, sense discrimination accuracy was relatively low. This motivated a digression into investigation of the influence of different distance measures; investigation of the geometry of the sense clusters in the LSA-based space through silhouette value analysis; investigation of sense discrimination accuracy as a function of the degree of supervision provided during model training; and investigation and comparison of sense discrimination in homonyms versus polysemes. Section three describes the investigation of optimal context size for word sense discrimination from 3 (1 word on each side of word) to 11 words (5 words on each side). Section 4 describes the use of Minimal Description Length (MDL) to determine the number of word senses. Section 5 provides a project summary. Appendix A provides a literature review and Appendix B provides a source code listing (not included in this published report).					
15. SUBJECT TERMS Latent Semantic Analysis, LSA, word sense disambiguation, lexical disambiguation, word sense frequency, word sense discrimination, k-means clustering, context-group discrimination, context vector, homonym, polyseme, polysemous, minimum description length, MDL, Double R Theory					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UNLIMITED	18. NUMBER OF PAGES 46	19a. NAME OF RESPONSIBLE PERSON Dr. Jerry Ball
a. REPORT UNCLASSIFIED	b. ABSTRACT UNCLASSIFIED	c. THIS PAGE UNCLASSIFIED			19b. TELEPHONE NUMBER (include area code) 480-988-6561

REPORT DOCUMENTATION PAGE

Form No. 1
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)
2. DATE COVERED FROM
3. DATE COVERED TO
4. CONTRACT NUMBER

5. REPORT TYPE
6. REPORT NUMBER

7. AUTHOR

8. GRANT NUMBER
9. PROGRAM ELEMENT NUMBER

10. PROJECT NUMBER

11. TASK NUMBER

12. WORK UNIT NUMBER

13. DATE

14. PERFORMING ORGANIZATION REPORT NUMBER

15. DISTRIBUTION STATEMENT (See Instructions for Authors)

16. DISTRIBUTION STATEMENT (See Instructions for Authors)
17. DISTRIBUTION STATEMENT (See Instructions for Authors)

18. DISTRIBUTION STATEMENT (See Instructions for Authors)

19. DISTRIBUTION STATEMENT (See Instructions for Authors)

20. DISTRIBUTION STATEMENT (See Instructions for Authors)

21. DISTRIBUTION STATEMENT (See Instructions for Authors)

22. DISTRIBUTION STATEMENT (See Instructions for Authors)

23. DISTRIBUTION STATEMENT (See Instructions for Authors)

24. DISTRIBUTION STATEMENT (See Instructions for Authors)

This page intentionally left blank.

25. DISTRIBUTION STATEMENT (See Instructions for Authors)

26. DISTRIBUTION STATEMENT (See Instructions for Authors)

27. DISTRIBUTION STATEMENT (See Instructions for Authors)

28. DISTRIBUTION STATEMENT (See Instructions for Authors)

29. DISTRIBUTION STATEMENT (See Instructions for Authors)

30. DISTRIBUTION STATEMENT (See Instructions for Authors)

31. DISTRIBUTION STATEMENT (See Instructions for Authors)

32. DISTRIBUTION STATEMENT (See Instructions for Authors)

33. DISTRIBUTION STATEMENT (See Instructions for Authors)

34. DISTRIBUTION STATEMENT (See Instructions for Authors)

35. DISTRIBUTION STATEMENT (See Instructions for Authors)

36. DISTRIBUTION STATEMENT (See Instructions for Authors)

37. DISTRIBUTION STATEMENT (See Instructions for Authors)

38. DISTRIBUTION STATEMENT (See Instructions for Authors)

39. DISTRIBUTION STATEMENT (See Instructions for Authors)

40. DISTRIBUTION STATEMENT (See Instructions for Authors)

41. DISTRIBUTION STATEMENT (See Instructions for Authors)

42. DISTRIBUTION STATEMENT (See Instructions for Authors)

43. DISTRIBUTION STATEMENT (See Instructions for Authors)

44. DISTRIBUTION STATEMENT (See Instructions for Authors)

45. DISTRIBUTION STATEMENT (See Instructions for Authors)

46. DISTRIBUTION STATEMENT (See Instructions for Authors)

47. DISTRIBUTION STATEMENT (See Instructions for Authors)

48. DISTRIBUTION STATEMENT (See Instructions for Authors)

49. DISTRIBUTION STATEMENT (See Instructions for Authors)

50. DISTRIBUTION STATEMENT (See Instructions for Authors)

51. DISTRIBUTION STATEMENT (See Instructions for Authors)

52. DISTRIBUTION STATEMENT (See Instructions for Authors)

53. DISTRIBUTION STATEMENT (See Instructions for Authors)

54. DISTRIBUTION STATEMENT (See Instructions for Authors)

Table of Contents

1	Introduction.....	6
2	Deliverable No 1: Investigation of optimal reduced dimensionality	8
3	Deliverable #2: Investigation of optimal context size.	18
4	Deliverable No. 3: The use of MDL to Determine the Number of Word Senses 19	
5	Summary	25
6	Acknowledgements.....	26
7	References.....	26
APPENDIX A.....		27
	Using LSA to Compute Word Sense Frequencies: Literature Review.....	27
1.	Introduction.....	27
2.	Double R Theory and WSD.....	27
3.	Word Sense Disambiguation.....	28
3.1	Corpus-based approaches to WSD.....	28
3.2	WSD and Psycholinguistics.....	31
4.	Latent Semantic Analysis	31
4.1	Vector Space Model (VSM)	32
4.2	Latent Semantic Analysis (LSA)	33
4.3	Extensions to LSA	36
5.	Model-based Clustering.....	38
5.1	The use of Regularization Methods to Determine the Number of Clusters (word senses).....	39
6.	Using LSA for word sense frequency estimation: Experimental Setup	41

The project described in this document is exploring the use of Latent Semantic Analysis (LSA) and statistical clustering techniques for automatically identifying word senses and for estimating word sense frequencies from application relevant corpora. The hypothesis is that LSA can be used to compute context vectors for ambiguous words that can be clustered together – with each cluster corresponding to a different sense of the word.

In addition to a relevant literature survey that was submitted earlier, and included here in Appendix A. and the source code in Appendix B, the deliverables for this project are:

1. Investigation of the effect of dimensionality of LSA-based representation on word sense discrimination.
2. Investigation of the effect of context size used in LSA on word sense discrimination.
3. Investigation of the use of regularization techniques in order to determine the number of senses automatically.

This report is organized as follows:

The next section includes short introduction of LSA, introduction of the context-group discrimination paradigm adapted in this project, and description of corpus used in the experiments. Section 2 describes the investigation of the effect of LSA dimensionality.

After the first set of experiments we were intrigued to understand why the sense discrimination accuracy was relatively low. This motivated the digression into investigation of influence of different distance measures; investigation of geometry of the sense clusters in the LSA-based space through silhouette value analysis; investigation of sense discrimination accuracy as a function of degree of supervision provided during model training; and investigation and comparison of sense discriminations in homonyms versus polysems. All these are described in section 2, followed by two sections describing the last two deliverables. Finally, we summarize the project in section 5.

1 Introduction

1.1 Latent Semantic Analysis

Latent semantic analysis is a mathematical technique used in natural language processing for finding complex and hidden semantic relations among words and the various contexts in which they are found (Landauer and Dumais, 1997; Landauer et al, 1998). LSA is based on the association of elements (words) with contexts. Similarity in word meanings is indicated by a corresponding similarity of the associated word contexts.

The starting point for LSA is the construction of a co-occurrence matrix, where the columns represent the different contexts in the corpus, and the rows represent the

different word tokens. Each entry ij in the matrix corresponds to the number of times the word token i appeared in context j . Often the co-occurrence matrix is normalized for document length and word entropy (Dumais, 1994).

The critical step of the LSA algorithm is the singular value decomposition (SVD) of the normalized co-occurrence matrix. If the matrices comprising the SVD are permuted such that the singular values are in decreasing order, they can be truncated to a much lower rank. According to Landauer and Dumais (1997), this dimensionality reduction step achieves the capturing of the relationship between words and passages and uncovers the relevant structural aspects while filtering out noise. The singular vectors correspond to the principal components, or axes of largest variance of the data, revealing the hidden abstract concepts of the semantic space. Words and documents can then be represented as a linear combination of these concepts.

Within the LSA framework discrete entities such as words and documents are mapped into the same continuous low-dimensional parameter space, revealing the underlying semantic structure of these entities and making it especially effective for a variety of machine-learning algorithms. Following successful application of LSA to information retrieval, other applications of the same methodology have been explored, including language modeling, word and document clustering, call routing, and semantic inference for spoken interface control (Bellegarda, 2005).

The goal of the work described here is to explore the use of LSA for unsupervised identification of word senses and for estimating word sense frequencies from application relevant corpora. In this paper we describe four sets of experiments aimed at investigating the tightness, separation, and purity of LSA-based clusters.

1.2 Context-Group Discrimination Paradigm

The basic idea of Schütze's (1998) context-group discrimination paradigm adopted in this investigation is to decode the sense of ambiguous words from their contextual similarity. The occurrences of ambiguous words represented by their context vectors are grouped into clusters consisting of contextually similar occurrences. The context vectors in our experiments are LSA-based representations of the documents in which the ambiguous words appear. Context vectors from the training portion of the corpus are grouped into clusters and the centroid of each cluster—the sense vector—is computed. Ambiguous words from the test portion of the corpus are disambiguated by finding the sense vector (cluster centroid) closest to the corresponding context vector representation. If sense labels are available for the ambiguous words in the corpus, sense vectors are given a label that corresponds to the most relevant sense in their cluster, and sense discrimination accuracy can be evaluated by computing the percentage of ambiguous words from the test portion that were mapped to the sense vector whose label corresponds to the ambiguous word's sense label.

1.3 Experimental Setup

We used the “line-hard-serve-interest” corpus (Leacock et al, 1993), including 1151 instances of 3 senses of the noun “Line” i.e. cord - 373, division - 374, and text - 404;

752 instances of 2 senses of the adjective “Hard”: i.e. difficult – 376, not yielding to pressure or easily penetrated – 376; 1292 instances of 2 senses of the verb “Serve”: i.e. serving a purpose, role or function or acting as – 853, and providing service 439; and 2113 instances of 3 senses of the noun “Interest”: i.e. readiness to give attention - 361, a share in a company or business – 500, money paid for the use of money -1252. All instances of each ambiguous word in the corpus were represented by the corresponding LSA context vectors. Since, in general, in this corpus the length of each document embedding an ambiguous word instance is relatively short (a paragraph of 33 words on average) we used the whole document as the context for computing the co-occurrence matrix.

In the experiments for deliverable No. 1 and No. 2 we assumed that the number of clusters corresponds to the number of different labeled senses in the corpus.

To study the influence that the proximity distance measure has on the disambiguation performance we considered three different distance measures:

1) **L1 or city-block distance defined as** $d(\vec{V}_i, \vec{V}_j) = \sum_k |V_{ik} - V_{jk}|$;

2) **L2 , or squared-Euclidean distance:**

$$d(\vec{V}_i, \vec{V}_j) = \sum_k (V_{ik} - V_{jk})^2 ; \text{ and}$$

3) **Cosine distance, defined as** $d(\vec{V}_i, \vec{V}_j) = 1 - \frac{\vec{V}_i \cdot \vec{V}_j}{|\vec{V}_i| |\vec{V}_j|}$.

2 Deliverable No 1: Investigation of optimal reduced dimensionality

2.1 Experiments.

To investigate the effectiveness of LSA representation for sense discrimination we conducted four sets of experiments described in the following subsections.

2.1.1 Sense-based Clusters: Supervised case.

Although our ultimate goal is the evaluation of the effectiveness of LSA for unsupervised word sense discrimination, in our first set of experiments sense labels were used directly to determine the sense-cluster centroids by averaging the training vectors with the same sense labels. This was done to evaluate the separation of the different senses in the LSA-based vector space. Ideally these sense-based clusters would be tight (the vectors in the cluster close to each other and close to centroid of the cluster), and well separated from each other; each cluster would be pure, i.e., consisting of vectors corresponding to words with the same sense, similarly to clusters shown in figure 1a. In this experimental setting

we evaluated the purity of sense based clusters, as opposed to clusters based on geometrical position of vectors. This corresponds to a supervised learning approach, since sense labels from the training portion of the corpus were used to define the clusters and their centroids. The relationship between supervised and unsupervised clustering is explained in the following subsection.

2.1.2 Supervised vs. unsupervised clustering.

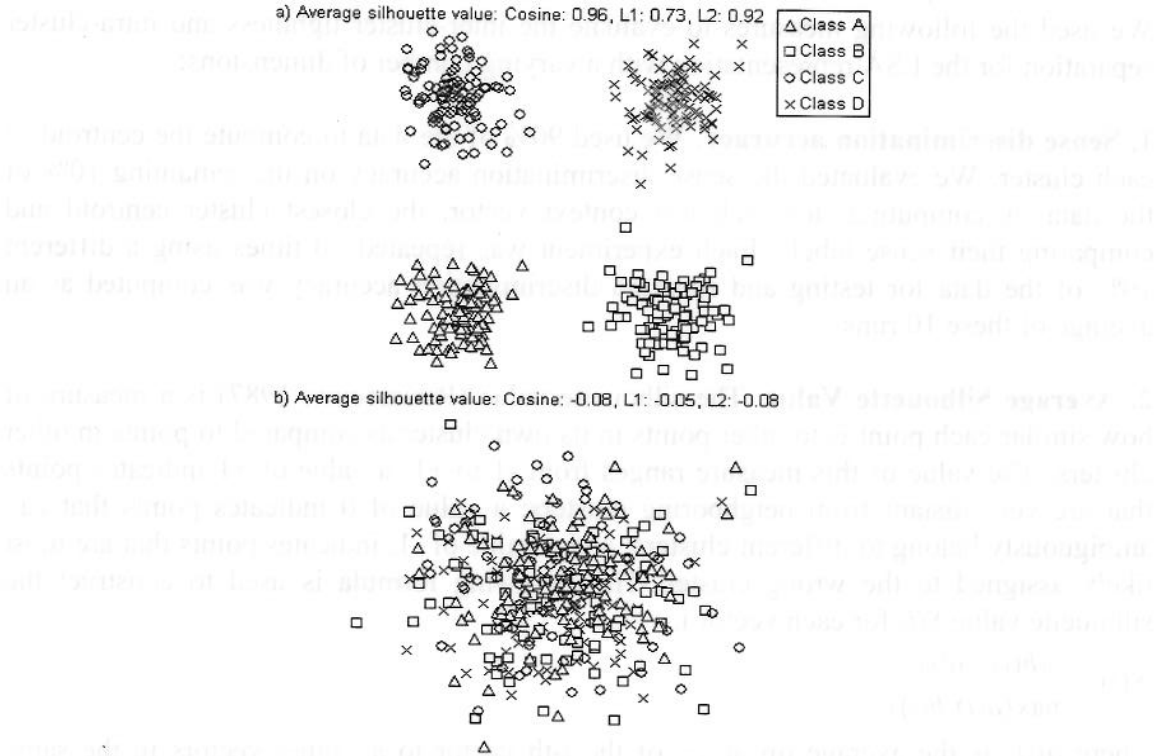


Figure 1: Extreme clustering scenarios

Figure 1 shows two extreme cases of clusters where we expect to get similar clustering performance for both supervised and unsupervised algorithms. The first case (figure 1a) corresponds to an ideal representation where the different senses form tight clusters that are well separated from each other. Here we can reasonably expect that unsupervised clustering approaches based on the geometrical relationships among vectors, such as K-means and EM, would perform as well as a supervised method in finding these clusters. The second extreme case, shown in figure 1b, corresponds to a poor vector space representation. Here the sets of vectors corresponding to the different senses of the word are overlapped, and we can expect low discrimination performance for both supervised and unsupervised methods. In general, supervised performance evaluated on the training set constitutes an upper-bound for the unsupervised clustering performance: The cluster centroids are, by definition, the points which have minimal average distance from each point in the same cluster (i.e. from the words in the training set with the same sense label). The centroids found by unsupervised clustering are instead based on geometric properties of all context vectors, regardless of their sense label.

The performance of unsupervised clustering, in practice, is often influenced by the parameters of each particular implementation, such as initialization, number of runs, stopping criterion, etc. Thus, by evaluating the performance of the supervised clustering one can obtain a computationally inexpensive upper-bound estimate of the performance of the unsupervised approach, which is independent of the implementation details.

2.1.3 Performance measures.

We used the following measures to evaluate the inter-cluster tightness and intra-cluster separation for the LSA representation with a varying number of dimensions:

1. Sense discrimination accuracy. We used 90% of the data to compute the centroid of each cluster. We evaluated the sense discrimination accuracy on the remaining 10% of the data by computing, for each test context vector, the closest cluster centroid and comparing their sense labels. Each experiment was repeated 10 times using a different 10% of the data for testing and reported discrimination accuracy was computed as an average of these 10 runs.

2. Average Silhouette Value. The silhouette value (Rousseeuw, 1987) is a measure of how similar each point is to other points in its own cluster as compared to points in other clusters. The value of this measure ranges from -1 to +1: a value of +1 indicates points that are very distant from neighboring clusters; a value of 0 indicates points that can ambiguously belong to different clusters; and a value of -1, indicates points that are most likely assigned to the wrong cluster. The following formula is used to construct the silhouette value $S(i)$ for each vector i :

$$S(i) = \frac{(b(i) - a(i))}{\max\{a(i), b(i)\}},$$

where $a(i)$ is the average distance of the i -th vector to all other vectors in the same cluster and $b(i)$ is the minimum of the average distance of the i -th vector to all vectors in the other cluster (i.e. the average distance between the points in the closest cluster among the other clusters). The overall average silhouette value is simply the average of the $S(i)$ for all points in the whole dataset. Since we investigate the quality of the clusters as a function of dimensionality of the underlying space, the silhouette value has the advantage of being normalized against the dimensionality of the vectors, as opposed to the *dispersion measure* D – i.e. the average distance of points in the cluster to cluster centroid – that is expected to increase with increased dimensionality. Average silhouette values for the two extreme cases discussed above are also shown in figure 1.

2.1.4 Results

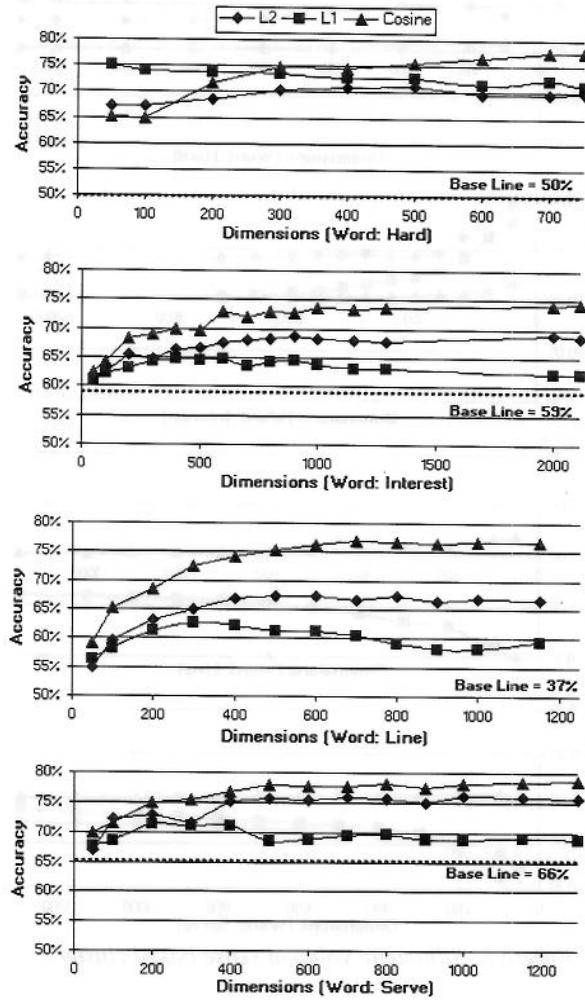


Figure 2: Accuracy of sense based clusters

The results of the first set of experiments with sense-based clusters are shown in Figures 2 and 3. Figure 2 plots the average discrimination accuracy of sense-based clusters as a function of LSA dimensionality for different distance measures, namely L2, L1 and cosine, for the 4 ambiguous words in the corpus. Note that the distance measure choice affects not only the classification of a point to the cluster, but also the computation of cluster centroids: For L2 and cosine measures the centroid is simply the average of vectors in the cluster, while for L1 the value of i -th dimension of the cluster centroid vector is the median of values of the i -th dimension of all the vectors in the cluster.

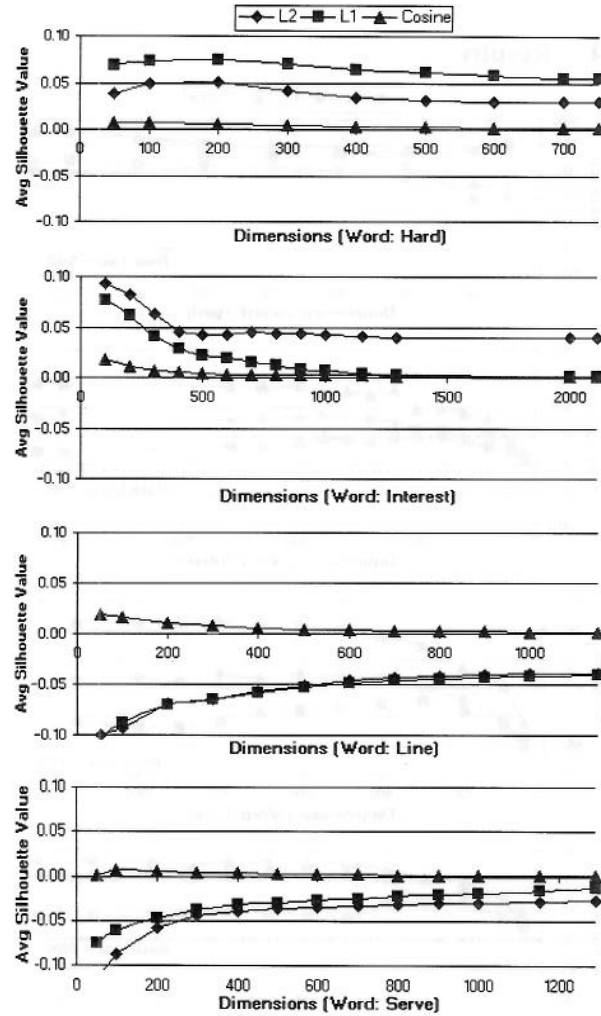


Figure 3: Silhouette value of sense based clusters

As shown by the sense discrimination results in Fig. 2, cosine distance, the most frequently used measure in LSA applications, has the best performance for 3 out of 4 words in the corpus. Only for the word “Hard” does L1 outperform cosine for low LSA dimensionality. For every number of dimensions examined, the average sense discrimination accuracy is significantly better than the baseline (computed as the relative percentage of the most frequent sense of each ambiguous word in the corpus).

Figure 3 shows the average silhouette values for the sense-based clusters as a function of the dimensionality of the underlying LSA-based vector representation for the 3 different distance metrics and for each of the 4 words in the corpus.

The average silhouette value is close to zero without significant variability with respect to the number of dimensions and distance measures. A possible explanation of these results, namely a combination of close to zero silhouette values and relatively good sense-discrimination accuracy, is that the context vectors in the LSA space are very close to the separation hyper plane, as is shown for 2-dimensional data in figure 4.

Average silhouette value: Cosine: 0.1821, L1: 0.0472, L2: 0.1560



Figure 4: Possible explanation for low silhouette values and good separation accuracy

2.2 Unsupervised Clustering.

In the second set of experiments we evaluated the clusters obtained in a non-supervised approach such as the K-Means algorithm. K-Means is a model-based partitionial clustering algorithm that is theoretically related to the EM algorithm (Alldrin et al, 2003). The K-Means algorithm finds K clusters by choosing K data points at random as the initial centroids. Each data point is then assigned to the closest centroid in order to form clusters by aggregation. Each centroid is then recomputed for each one of the new clusters. This process is iterated until no data points are reassigned to different clusters. It can be shown that K-Means performs a local minimization of the *dispersion*, D , i.e. the sum of the distances of each point from the centroid of the cluster to which it belongs.

2.2.1 Results

Figure 5 summarizes the results of the second set of experiments with clusters obtained by the K-means algorithm. In this set of experiments, only the cosine distance measure was used. Figure 5 shows the discrimination accuracy as a function of the dimensionality of the LSA representation. Each plot corresponds to one of the 4 different words in the corpus. Curves labeled K-0 and K-1, shown here for comparison, report the results of the first set of experiments: K-1 is the accuracy with the cosine distance from Figure 1; and K-0 is the discrimination accuracy as evaluated on the training set. The word sense discrimination accuracy estimated in this way is an upper bound on the word sense discrimination performance of unsupervised clustering such as K-means or EM, as previously explained in section 4.1.1. For each word in the corpus and for each number of dimensions we ran 10 cycles of K-means starting from random centroids. The scattered points labeled as K-3 describe the accuracy for those experiments. The single point for each number of dimensions labeled as K-4 represents the accuracy of the cluster with the smallest dispersion D . To estimate the performance for the best possible initialization of K-means (curve K-2), we set initial cluster centroids as the sense-clusters centroids.

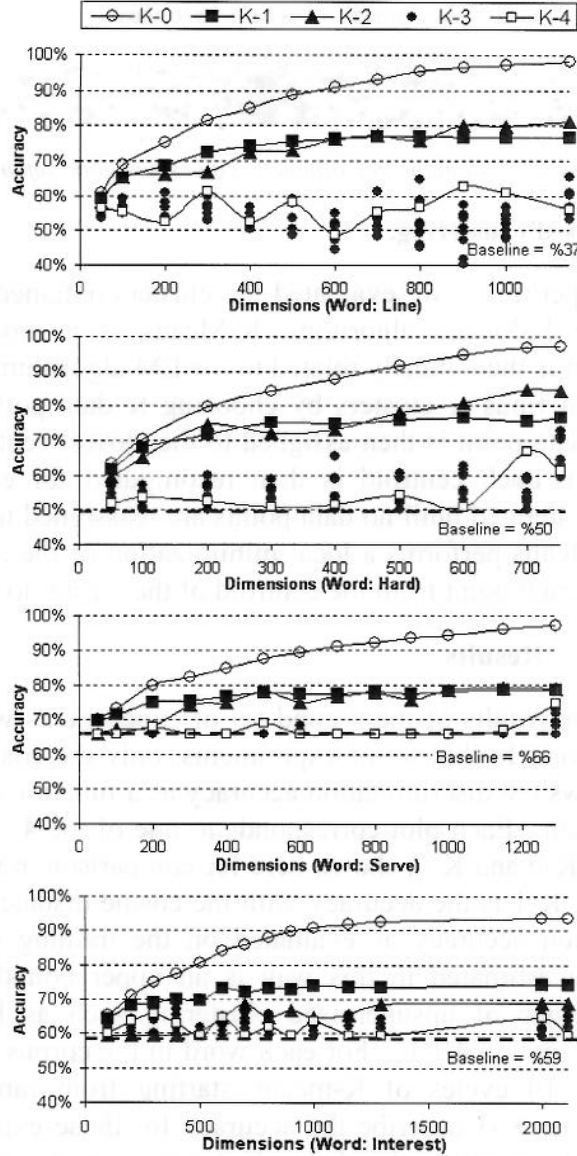


Figure 5: Results with Unsupervised Clustering

2.3 Semi - Supervised Clustering

The best performance of K-means shown in figure 5 by the curve labeled K-2 was obtained by selecting the best initialization: the initial cluster centroids were computed as sense-based clusters' centroids. Notice that the discrimination accuracy in this case is very close to the one attained by supervised clustering, described by curve K-1 in figure 5. This, of course, is not surprising, since we used sense labels to find a good starting point for K-means, just like in the supervised case. However, we can see the two types of clustering: 1) unsupervised clustering initialized from random points (K-3 in figure 5); and 2) clustering initialized in a 'supervised' manner (curve K-2 in figure 5) as two extreme points in a continuum of varying degrees of supervision. Thus, in the

experiments reported in this subsection we measure the sense discrimination performance as a function of *degree of supervision*.

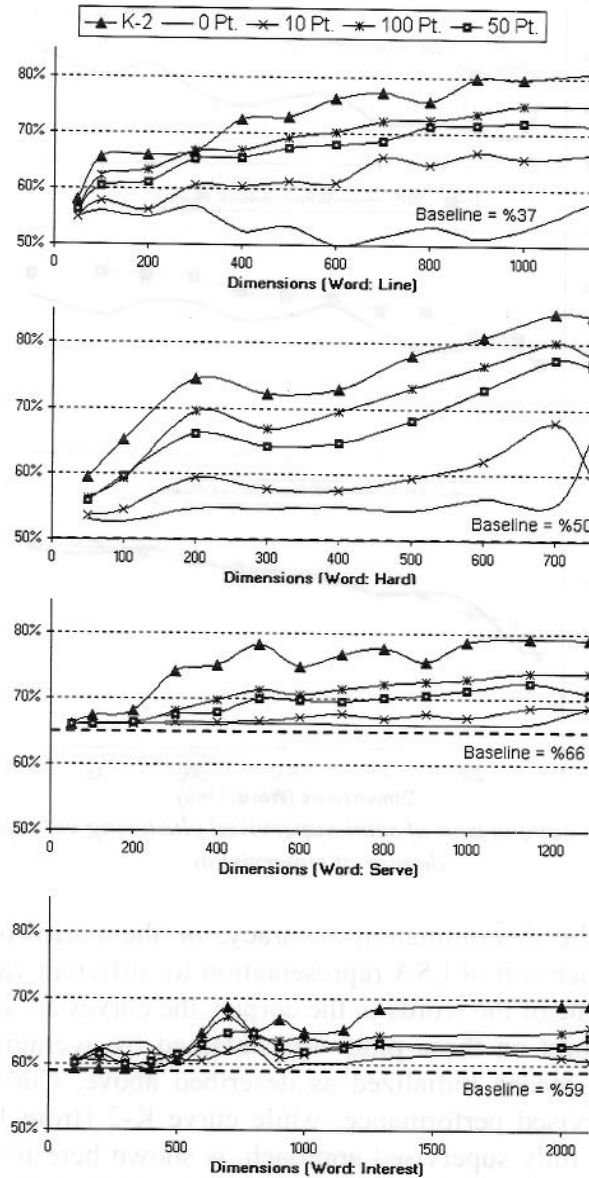


Figure 6: Semi-supervised Clustering

Here we use supervision, similarly to the experiment that yielded the curve K2 in figure 5, to find the initialization points (initial sense cluster centroids) for the K-Means algorithm. We refer to this as semi-supervised clustering: the idea here is to use two sets of data: a small *initialization set* of labeled data to determine the initial cluster centroids for K-means, and a large unlabeled set which is used during the successive iterations of K-means to produce the ultimate clusters. We define the degree of supervision (DS) as the number of labeled instances from each word-sense that were used to compute the initial centroids. For example, for DS = 1, we chose, randomly, one instance for each sense to serve as initial cluster centroid for that sense, while for DS = 10, we used 10 random instances.

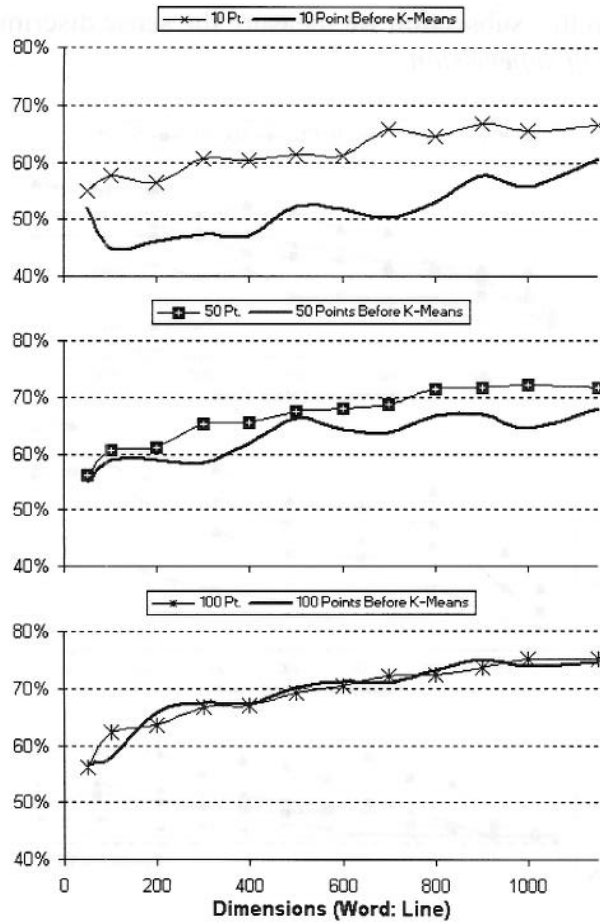


Figure 7: performance comparison of semi-supervised clustering vs. supervised for a given degree of supervision.

Figure 6 describes the discrimination accuracy, or the purity of the clusters, as a function of reduced dimension of LSA representation for different values of DS. In each plot, corresponding to one of the words in the corpus, the curves are labeled by the degree of supervision. Each point on these plots was obtained by averaging the results of 10 independent runs of K-means initialized as described above. Curve labeled by DS=0 corresponds to unsupervised performance, while curve K-2 (from figure 5), describing the performance of the fully supervised approach, is shown here for reference. The five curves in each plot show the performance monotonically increasing with the degree of supervision.

We can consider the semi-supervised learning to be performed in two stages: the first one is supervised, where labeled data is used to compute an estimate of cluster centroid; and the second one, unsupervised, where clustering is performed on larger set of unlabeled data using the estimates obtained in the first stage as initialization points. To measure the value of the second, unsupervised learning stage for sense discrimination accuracy we compared the sense discrimination accuracy of the two stage semi-supervised algorithm (from figure 6) to the one obtained by clusters whose centroids are the estimates computed in the first, supervised, stage.

As is shown in Figure 7 for the word “Line”, while the unsupervised clustering of the second stage improves the discrimination performance, the relative improvement diminishes as the size of the labeled data used in the supervised stage increases.

These results indicate that:

- a) The performance of the non-supervised approach degrades significantly with respect to the performance of the supervised method. Except for the word *line*, the discrimination accuracy of the most compact cluster (K-4) is close to the baseline.
- b) The performance of K-means is highly dependent on the initialization as shown by a wide scatter of points (K-3).
- c) There is no correlation between the compactness of the clusters (as measured by dispersion D) obtained by K-means and their discrimination accuracy, or, in other words, the most compact clustering (K-4) does not necessarily yield the best discrimination result.
- d) Figures 2, 3, 5 do not show any pronounced maximum value for any range of LSA number of dimensions. These results suggest that the reduction of dimensionality does not increase the sense discrimination power of the LSA representation, but only makes the computation more efficient and thus enables processing of much larger corpora.

2.4 Polysems vs. Homonyms

Ambiguous words in the line-hard-serve-interest corpus are examples of polysems. A **polyseme** is a word with multiple meanings related to each other. On contrast, homonyms are homophonous words with unrelated meanings. The noun *bank*, as a *financial institution* and *bank as river-bank* are examples of homonyms. Disambiguation of polysems is a difficult problem even for humans (Kilgarriff, 1997), and even more so for automatic sense discrimination algorithms. To evaluate the performance of LSA-based clusters on homonyms and to contrast it to the performance on polysems we conducted a third set of experiments described in this subsection.

We followed the evaluation paradigm proposed in (Schütze, 1998), where artificial or pseudowords were used as a convenient means of testing disambiguation. Two or more words, in our case, *line*, *hard*, *serve* and *interest*, are conflated into a new type: *line-hard-serve-interest*. All occurrences of either word in the corpus are then replaced by the new pseudoword. It is easy to evaluate disambiguation performance for pseudowords since one can go back to the original text to decide whether a correct decision was made.

Figure 8 presents the sense discrimination results for this set of experiments for a pseudoword corpus consisting of 1000 random instances of each of the 4 original words substituted with the pseudoword type. Curves and points labeled K-0 to K-4 in figure 4 are in full analogy to Figure 5. The results in figure 8 show that not only the discrimination performance for pseudowords is much above the baseline, but also that the performance of the unsupervised method is very close to that of the supervised one: curve K-4 obtained by finding the most compact cluster in 10 runs of K-means initiated from random points is very close to its upper bound K-2 and to the supervised performance K-1, in contrast to the results for polysems in Figure 5.

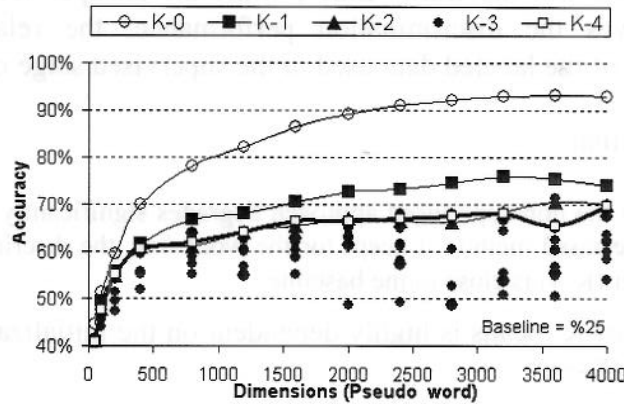


Figure 8: Discrimination accuracy for homonyms

3 Deliverable #2: Investigation of optimal context size.

In this section we describe experiments aimed at evaluating the optimal LSA context size for word – sense discrimination.

Figures 9 -11 present the sense discrimination results for a pseudoword corpus consisting of 1000 random instances of each of the 4 original words substituted with the pseudoword type, the same corpus used in section 2.4. Curves and points labeled K-0 to K-4 in these figures are in full analogy to Figures 5 and 8.

Figure 9 presents discrimination results for LSA representation of pseudowords, when context window of 3 words centered on the word in question is used.

In Figure 10, context window of 7 words (3 on each side) and in Figure 11 – a window of 11 words (5 on each side) centered on the current word was used.

The reference context – entire document – was used in experiments described in section 2.4 and represented by figure 8.

Note that the maximal dimension of LSA representation also depends on the context size, since it is the rank of the co-occurrence matrix.

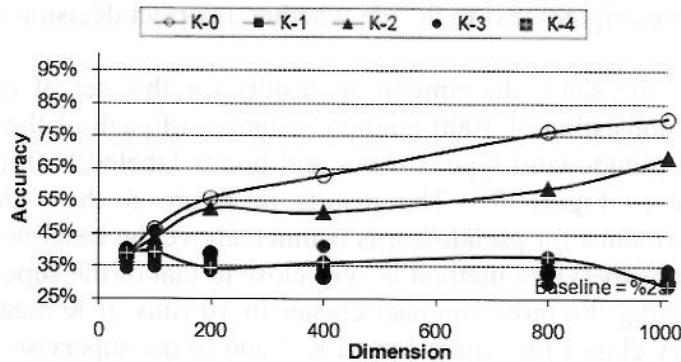


Figure 9: Discrimination accuracy for context window of 3 words (one on each side)

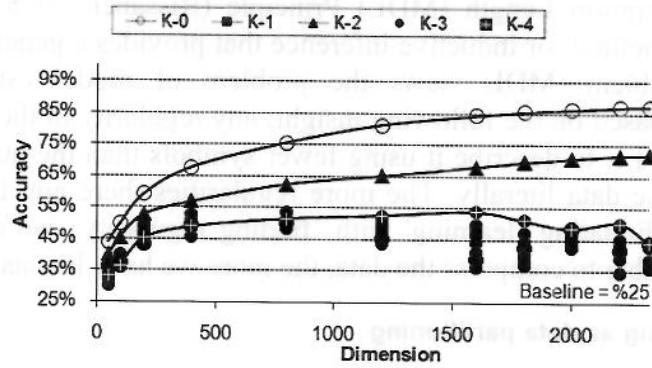


Figure 10: Discrimination accuracy for context window of 7 words (3 on each side)

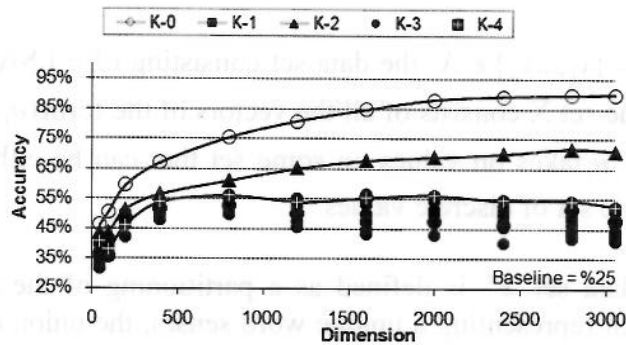


Figure 11: Discrimination accuracy for context window of 11 words (5 on each side)

Comparison of figures 8 – 11 indicates that the best unsupervised discrimination results (curve K-4) are obtained for LSA representation based on full document context; moreover, the results improve monotonically when the context size grows. For reference, in these 4000 documents that constitute the homonym corpus, the average document length (full context size) is 18 (when excluding stop words).

4 Deliverable No. 3: The use of MDL to Determine the Number of Word Senses

The problem of unsupervised clustering when the number of clusters is not known is ill-posed. The error of fitting to the data can only be reduced by increasing the number of clusters: the best fit for the data would be a trivial clustering were the number of clusters is equal to the number of vectors in our data set. For WSD that means a different sense for every occurrence of a word. Of course, this trivial clustering does not provide any information on the nature of the data. To find automatically both the best number of cluster and meaningful clustering, regularization methods can be used. Regularization is a method of imposing additional conditions for solving inverse problems with optimization methods. With regularization, the objective function that the clustering algorithms are trying to optimize measures not only the fit of the model to the data (this fit can only be made better by adding clusters), but also a penalty term that measures the complexity of the model (this penalty is smaller for simpler models with fewer clusters).

In particular, in this project we used Minimal Description Length Principle.

The Minimum Description Length (MDL) Principle (Rissanen 1978; Rissanen 1987; Rissanen 1996) is a method for inductive inference that provides a generic solution to the model selection problem. MDL views the problem of model estimation as data compression, and is based on the following insight: any regularity in the data can be used to compress the data, i.e. to describe it using fewer symbols than the number of symbols needed to describe the data literally. The more regularities there are, the more the data can be compressed. Equating 'learning' with 'finding regularity', we can therefore say that the more we are able to compress the data, the more we have learned about the data.

4.1 Clustering as data partitioning

In the following we will use notation and introductory explanations from (Kontkanen et al, 2005).

Let us denote by $x^n = \{x_1, \dots, x_n\} \in X$ the data set consisting of n LSA representations of ambiguous words. The set X consists of all the vectors of the form (a_1, \dots, a_m) , where each variable (or attribute) a_i takes on values on some set that can be either a continuum of real numbers, or a finite set of discrete values.

A clustering of the data set x^n is defined as a partitioning of the data into mutually exclusive subsets (each representing a unique word sense), the union of which forms the data set. The number of subsets is a priori unknown. The clustering problem is the task to determine the number of subsets, and to decide to which cluster each data vector belongs.

Formally, we can notate a clustering by using a clustering vector $y^n = \{y_1, \dots, y_n\}$, where y_i denotes the index of the cluster to which the data vector x_i is assigned to. The number of clusters K is implicitly defined in the clustering vector, as it can be determined by counting the number of different values appearing in y^n . Hence the clustering problem is now to find from all the $y^n \in \Omega$ the optimal clustering \hat{y}^n .

4.2 Model Class

Consider a set $\Theta \in R^d$. A class of parametric distributions indexed by the elements of Θ is called a model class. That is, a model class M is defined as the set $M = \{P(\cdot | \theta : \theta \in \Theta)\}$.

In the following, we use the simple finite mixture as the model class. In this case, the probability of a single data vector is given by

$$4) \quad P(x | \theta, M_K) = \sum_{k=1}^K P(x | y = k, \theta, M_K) P(y = k | \theta, M_K),$$

so that a parametric model θ is a weighted mixture of K component models $\theta_1, \dots, \theta_K$, each determining the local parameters $P(x | y = k, \theta, M_K)$ and $P(y = k | \theta, M_K)$.

Furthermore, as is usually done in mixture modeling, we assume that the variables $\{a_1, \dots, a_m\}$ are locally (conditionally) independent:

$$5) \quad P(x | y = k, \theta, M_K) = \prod_{i=1}^m P(a_i | y = k, \theta, M_K)$$

The above assumes that the parameter K is fixed. Since the number of clusters is bounded by the size of the available data set, in the following we consider the union of model classes M_1, \dots, M_n .

4.3 Clustering and MDL

Our optimality criterion for clustering is based on information-theoretical arguments, in particular on the Minimum Description Length (MDL) principle (Rissanen 1978; Rissanen 1987; Rissanen 1996).

Intuitively, the MDL principle aims at finding the shortest possible encoding for the data, in other words the goal is to find the most compressed representation of the data. Compression is possible by exploiting underlying regularities found in the data — the more regularities found, the higher the compression rate. Consequently, the MDL optimal encoding has found all the available regularities in the data; if there would be an “unused” regularity, this could be used for compressing the data even further.

How to formalize the above intuitively motivated MDL approach for clustering?

To proceed, let's restate the well-known fact from information theory about the fundamental relationship between codes and probability distributions: for every probability distribution P , there exists a code with a code length $-\log P(x)$ for all the data vectors x , and for each code there is probability distribution P such that $-\log P(x)$ yields the code length for data vector x (see [Cover and Thomas 1991]). This means that we can compress a cluster efficiently, if our model class yields a high probability for that set of data. Globally this means that we can compress the full data set x^n efficiently if $P(x^n | M)$ is high, where M is the class of models consisting of all possible clustering with number of clusters $K \leq n$.

Now we can restate the clustering problem as finding number of clusters K , the model $\theta \in M_K$ and a clustering vector $y^n = \{y_1, \dots, y_n\}$ that maximizes the probability (or minimizes the corresponding length of code)

$$6) \quad \hat{y}^n, \hat{\theta}, K = \arg \max P(y^n, x^n | K, \theta) \cdot P(K, \theta) = \\ = \arg \max P(y^n | K, \theta) \cdot P(x^n | y^n, K, \theta) \cdot P(K, \theta)$$

Finding the best model for clustering results therefore in a three part code: the first part is the code needed to transmit the membership function given the data and a particular model; the second part is the code required to transmit the data given the model and the membership; and the third one is the code to transmit the parameters of the model itself:

$$7) \hat{y}^n, \hat{\theta}, \hat{K} = \arg \max [\log P(y^n | K, \theta) + \log P(x^n | y^n, K, \theta) + \log P(K, \theta)]$$

In the following section we describe the models and the assumptions needed for computation of the terms of the equation above in our case.

4.4 Applying MDL Principle to LSA vector clustering.

4.4.1 The first term: code length for the membership function.

The length of code to transmit the membership function is: $\log P(y^n | K, \theta)$, where y^n is the membership assignment vector of length n , where each component of this vector is an integer between 1 and K specifying the identity of the cluster the appropriate data vector belongs to. We will assume here that all possible clusterings are equiprobable, (with probability of $1/K^n$) resulting in

$$8) \log P(y^n | K, \theta) = -n \log K$$

4.4.2 The second term: code length for transmitting the data.

Assumption 1: The data is coming from an equiprobable mixture of normal distributions. Since the membership of each data vector is given, it identifies the corresponding normal density for the data vector:

$$9) p(x | y = i, K, \theta) = N(C_i, I),$$

where C_i denotes i -th cluster centroid, and I is the identity covariance matrix.

This assumption is made not only for mathematical convenience, but mainly since maximization of (9) corresponds to the K-means algorithm we use in this project for clustering with Euclidian distance measure (Eq. 2).

Note that (9) is a probability density, while the corresponding term in Eq.(7) is a probability. By its nature, our data (the LSA-based vectors) are real-valued vectors, and their *probability* (as opposed to probability density) is not defined. Therefore in order to use the MDL principle we need to quantize the data to make it discreet.

Assumption 2: We quantize the data into small axis parallel hyper cubes with a side length of Δx . We assume that Δx is small enough so that the integral of normal density of Eq. (9) -the probability mass- can be approximated by

$$10) P(x | y = i, K, \theta) \cong (\Delta x)^d p(x | y = i, K, \theta) = \left(\frac{\Delta x}{\sqrt{2\pi}}\right)^d \exp\left(-\frac{1}{2} \|x - C_i\|^2\right),$$

where $\|\cdot\|$ denotes vector norm. From this, the corresponding code length (term 2 of Eq. 7) is

$$\begin{aligned}
11) \quad \log P(x^n | y^n, K, \theta) &= d \cdot n(\log \Delta x - \frac{1}{2} \log 2\pi) - \frac{1}{2} \sum_{i=1}^n |x_i - C_{y_i}|^2 = \\
&= \text{const} - \frac{1}{2} D(x^n, y^n, \theta),
\end{aligned}$$

where $D(x^n, y^n, \theta)$ is the dispersion.

4.4.3 The third term: code length for transmitting the model.

In our case, the model parameters θ are a set of K centroid vectors. We assume that the centroids are independent and uniformly distributed on the discrete hypercube defined by Δx . This hypercube is the smallest that still contains all the data, i.e., the a -th dimension of the hypercube has the length of $x_a^{\max} - x_a^{\min}$, where x_a^{\max} and x_a^{\min} are the maximum and the minimum values of the a -th dimension of all LSA data vectors, correspondingly.

The uniform probability over this discrete cube is

$$12) P(C) = 1 / \prod_{a=1}^d (x_a^{\max} - x_a^{\min}) / \Delta x,$$

and the corresponding code length is

$$13) \log P(K, \theta) = \log \prod_{k=1}^K P(C_k) = K \cdot d \cdot \log \Delta x - K \cdot \sum_{a=1}^d \log(x_a^{\max} - x_a^{\min}).$$

4.5 Experimental Evaluation

Now we are ready to rewrite Eq. 7 using the terms of equations 8, 11 and 13.

$$\begin{aligned}
14) \quad \hat{y}^n, \hat{\theta}, \hat{K} &= \arg \max [-\frac{1}{2} D(x^n, y^n, \theta) - (n \log K - K \cdot d \cdot \log \Delta x + K \cdot \sum_{a=1}^d \log(x_a^{\max} - x_a^{\min}))] = \\
&= \arg \max [L(y^n, \theta, K)].
\end{aligned}$$

In Eq. 14 we removed the terms that are constant with respect to y^n, θ and K . Note, that as expected, the objective function $L(y^n, \theta, K)$ includes now 2 terms: the first one is proportional to the dispersion; and the second is a penalty term that depends upon the model complexity – the number of clusters K .

We performed maximization of Eq. 14 in two stages. For each possible value of number of clusters K we maximized $L(y^n, \theta, K)$ of Eq. 14 over parameters and membership function. This maximization is equivalent to minimizing the dispersion, since the other terms are constant for a given K . The minimization was performed by K-means algorithm by simultaneously finding the cluster centroids (parameters) and assigning the data to clusters (membership function). Since K-means finds a local minimum that is dependent on its initialization, for each value of K under consideration we averaged the resulting minimal dispersion over 10 runs of K-means.

In our experiments we used the pseudo words corpus of 4000 documents described in the pervious section ($n = 4000$), with LSA dimensionality set to 50 ($d=50$). Since our model assumes that the data dimensions are independent, we applied whitening transformation to the raw data, by performing a linear transformation that guarantees that the data covariance matrix is identity.

Figures 12, 13, and 14 describe the results of experiments for the number of clusters K taking 21 values from 2 to 400 clusters.

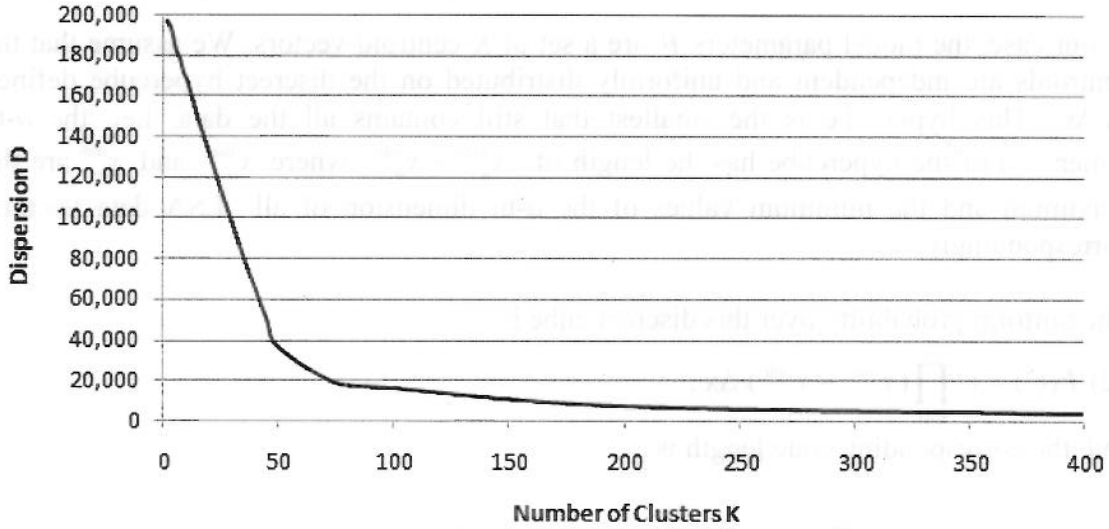


Figure 12: Dispersion D as a function of number of clusters

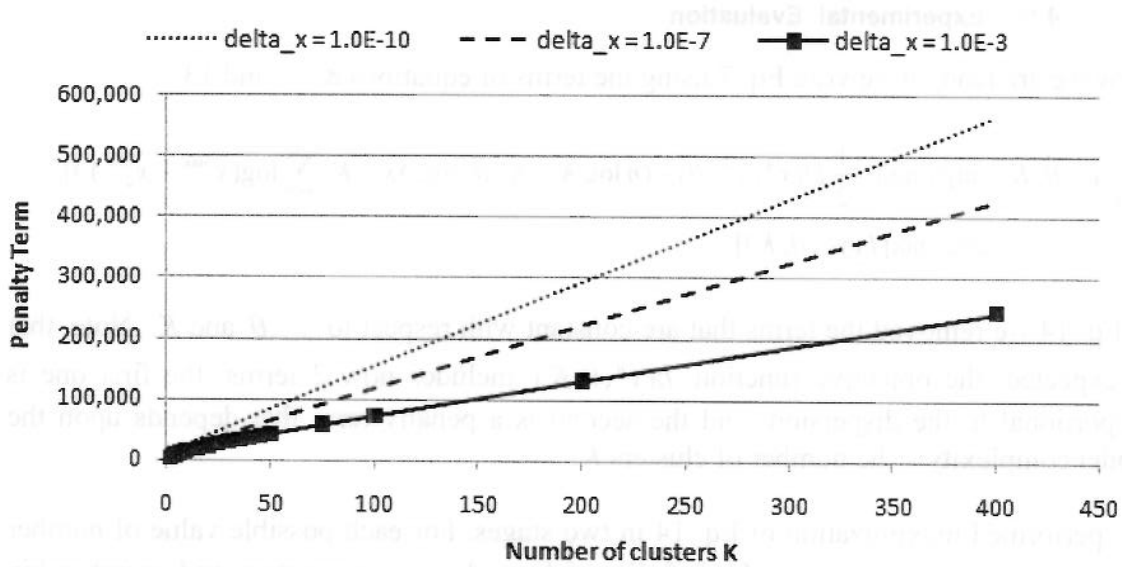


Figure 13: The penalty term as a function of number of clusters

In Figure 12 the averaged minimal dispersion D is plotted as a function of number of clusters. Note, that as expected, this curve is monotonically non-increasing, since by adding clusters we cannot make the minimal dispersion larger. In Figure 13 we plot the K -dependent penalty term

$n \log K - K \cdot d \cdot \log \Delta x + K \cdot \sum_{a=1}^d \log(x_a^{\max} - x_a^{\min})$ for several values of Δx , ranging from $1.0E-10$ to $1.0E-3$.

Figure 14 represents the sum $-\frac{D}{2} - [n \log K - K \cdot d \cdot \log \Delta x + K \cdot \sum_{a=1}^d \log(x_a^{\max} - x_a^{\min})]$ of Equation 14 for the different values of Δx . The maximum in these curves corresponds to the numbers of clusters that is optimal according to MDL principle.

Our results indicate that the value of optimal \hat{K} is quite insensitive to the choice of Δx . For two Δx values of $1.0E-3$ and $1.0E-7$ the optimal numbers of clusters \hat{K} was 50; further decreasing Δx to $1.0E-10$ changes \hat{K} only slightly to $\hat{K} = 45$. However, in all cases, the found number of sense clusters does not correspond to the number of pseudo-homonyms we used to construct the corpus which was 4.

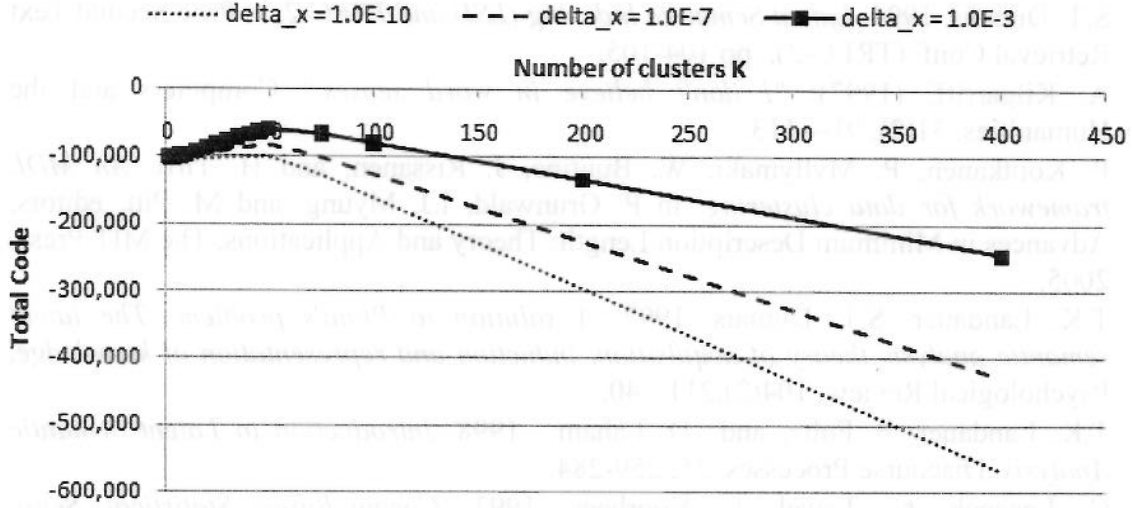


Figure 14: The total code length (Eq. 14) as a function of number of clusters. Maximum point of each curve corresponds to the optimal number of clusters K .

5 Summary

This report summarizes the research performed for exploring the use of Latent Semantic Analysis (LSA) and statistical clustering techniques for automatically identifying word senses and for estimating word sense frequencies from application relevant corpora.

Three deliverables are described in the report, namely, investigation of the effect of dimensionality of LSA-based representation on word sense discrimination; investigation of the effect of context size used in latent semantic analysis on word sense discrimination; and investigation of the use of regularization techniques in order to determine the number of senses automatically.

6 Acknowledgements

The project described above is supported by grant “Using LSA to Compute Word Sense Frequencies” from Air Force Research Laboratories Award No FA8650-05-1-6637 . Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the AFRL.

7 References

- N. Alldrin, A. Smith and D. Turnbull, 2003, “Clustering with EM and K-Means”, University of San Diego, California, Tech Report
- J.R. Bellegarda. 2005. *Latent Semantic Mapping*, IEEE Signal Processing Magazine, 22(5):70-80.
- Cover, T. and J. Thomas (1991). *Elements of Information Theory*. New York, NY: John Wiley & Sons.
- S.T. Dumais. 1994. *Latent Semantic Indexing (LSI) and TREC-2*, in Proc Second Text Retrieval Conf. (TREC-2), pp 104-105.
- A. Kilgarriff, (1997): “*I don't believe in word senses.*” Computers and the Humanities, 31(2), 91—113.
- P. Kontkanen, P. Myllymaki, W. Buntine, J. Rissanen, and H. Tirri. *An MDL framework for data clustering*. In P. Grunwald, I.J. Myung, and M. Pitt, editors, *Advances in Minimum Description Length: Theory and Applications*. The MIT Press, 2005.
- T.K. Landauer, S.T. Dumais. 1997. *A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge*, Psychological Review, 104(2):211-240.
- T.K. Landauer, P. Foltz, and D. Laham. 1998. *Introduction to Latent Semantic Analysis*. Discourse Processes, 25, 259-284.
- C. Leacock, G. Towel, E. Voorhees. 1993. *Corpus-Based Statistical Sense Resolution*, Proceedings of the ARPA Workshop on Human Language Technology.
- Rissanen, J. 1978. *Modeling by shortest data description*. Automatica 14, 445– 471.
- Rissanen, J. 1987. *Stochastic complexity*. Journal of the Royal Statistical Society 49(3), 223–239 and 252–265
- Rissanen, J. 1996. *Fisher information and stochastic complexity*. IEEE Transactions on Information Theory 42(1), 40–47.
- P.J. Rousseeuw. 1987. *Silhouettes: a graphical aid to the interpretation and validation of cluster analysis*. Journal of Computational and Applied Mathematics. 20. 53-65.
- H. Schütze. 1998. *Automatic Word Sense Discrimination*, Journal of Computational Linguistics, Vol. 24, Number 2.

Using LSA to Compute Word Sense Frequencies: Literature Review

1. Introduction

The goal of this project is to study Latent Semantic Analysis (LSA) based methods for estimation of word sense frequencies. Existing word frequency measures are based on relative frequency of word appearance in a corpus. To estimate word-sense frequency for a particular word token W we need to be able to tag all instances of W in the corpus by their corresponding sense. Hence, the problem of word-sense frequency estimation is closely related to the problem of Word Sense Disambiguation (WSD).

Our approach in this project is based on word sense frequency estimation by unsupervised model-based clustering of polysemous words contexts described by their LSA vector representation. Given such clustering, the word sense frequency distribution can be computed as a relative count of instances in each cluster.

This report surveys the literature pertaining to this project. It is organized as follows:

Section 2 describes the motivation for this project, namely the need for word sense frequency estimation in Double R Model. Section 3 focuses on exciting methodology for word sense disambiguation, including supervised, dictionary-based, and unsupervised methods. Section 4 describes LSA methodology and its extensions. Section 5 describes clustering algorithms and regularization methods that can be applied to automatically determine the number of clusters. Section 6 concludes with experimental setup for the project.

2. Double R Theory and WSD

Double R Model (Referential and Relational Model) (Ball, 2003; 2004) is a model of natural language. Double R Model adopts a cognitively plausible approach to modeling language comprehension while attempting to support the development of large-scale models. *Double R Grammar* is the Cognitive Linguistic theory of the grammatical encoding of referential and relational meaning underlying Double R Model, and *Double R Process* is the theory of language processing underlying Double R Model, constituting a cognitively plausible processing mechanism for constructing integrated representations of referential and relational meaning.

A computational implementation of Double R Model, is based on ACT-R [Anderson & Lebiere, 1998; Anderson et als, in press]. ACT-R is a cognitive architecture and modeling environment for the development of computational cognitive models that has been used extensively in the modeling of higher-level cognitive processes (see the ACT-R web site at <http://act-r.psy.cmu.edu/> for an extensive list of models and publications). ACT-R includes symbolic production and declarative memory systems integrated with sub-symbolic production selection and spreading activation and decay mechanisms. A key requirement of Double R Model is the ability to disambiguate the meaning of polysemous

words. In ACT-R terms, this corresponds to retrieving the appropriate sense of a word (using the activation equation,) given the current input word, the prior context (i.e. prior words) and the history of use of the various senses of the input word. Here the frequency of occurrence of a word sense translates into base level activation (Bi) of the word sense and context translates into activation spread from the prior input ($\sum_j W_j \cdot S_{ji}$). These two terms combine to determine the total activation of the word sense (Ai) in context. Typically, the most highly activated word sense will be selected as the appropriate word sense during processing.

The need to estimate frequency of occurrence of a word sense for computing base level activation (Bi) motivates this project.

3. Word Sense Disambiguation

In general terms, Word Sense Disambiguation involves the association of a given word in a text or discourse with a definition or meaning (sense) which is distinguishable from other meanings potentially attributable to that word (Ide and Véronis, 1998). Words in any language can have multiple meanings and their correct meaning is determined in context, that is, in presence of co-occurrence with other words (i.e., co-occurrence). Sense disambiguation is usually regarded as an “intermediate task” (Wilks and Stevenson, 1996) which is not an end in itself, but rather is necessary at one level or another to accomplish most natural language processing tasks, including natural language understanding, information retrieval (such as Internet search engines) and machine translation.

3.1 Corpus-based approaches to WSD.

The WSD task necessarily involves two steps: (1) the determination of all the different senses for every word relevant (at least) to the text or discourse under consideration; and (2) a means to assign each occurrence of a word to the appropriate sense.

Much recent work on WSD relies on pre-defined senses for step (1). The precise definition of a sense is, however, a matter of considerable debate within the Community (Kilgarriff, 1997). To avoid the precise definition of senses in Barwise and Perry’s (1953) situation semantics was proposed, where the sense or senses of a word are seen as an abstraction of the role that it plays systematically in the discourse. Schütze (1992, 1993) proposed a method that avoids the problem of sense distinction altogether: he creates sense clusters from a corpus rather than rely on a pre-established sense list. In this survey we will classify the methodologies applied to WSD according to the amount of supervision they require.

3.1.1 Supervised Methods

Supervised WSD seeks to learn classifiers that can assign senses to words in text using machine learning techniques. These classifiers are trained on manually sense tagged corpora. In general these classifiers are most suitable for lexical sample or target word disambiguation, where all the occurrences of a given word in a text are assigned a sense.

We can classify the supervised methods into two main approaches, namely, Naïve Bayes, and Information-theoretic.

Naïve Bayes: Gale et al. (1993) approached supervised WSD by looking at the words around an ambiguous word in a large context window. Each content word contributes potentially useful information about which sense of the ambiguous word is likely to be used with it. The classifier does no feature selection. Instead, it combines the evidence from all features, according to

Bayes decision rule:

Decide s' if for $s_k \neq s'$, where $P(s_k | C)$ is computed by Bayes' Rule.

Naïve Bayes classifier assumes that the words are independent (this is also known as “bag of words” model): $P(s_k | C) = P(\{v_j | v_j \text{ in } C\} | s_k) = \prod_{v_j \text{ in } C} P(v_j | s_k)$ The Naive Bayes assumption is incorrect in the context of text processing, but it is useful.

When a new word needs to be disambiguated, for each sense of that ambiguous word, a score for the context is being calculated by summing up the scores of every word and disambiguation is done by summing up scores of all the words in context and choosing the sense with highest score. Other works that use Naïve Bayes approach include (Mooney, 1996; Ng, 1997; Leacock et al., 1998).

Information Theory: In (Brown et al., 1991) an algorithm was proposed that finds a single contextual feature that reliably indicates which sense of the ambiguous word is being used. The *Flip-Flop* algorithm disambiguates between the different senses of a word using the mutual information as a measure.

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

This algorithm works by searching for a partition of senses that maximizes the mutual information. The algorithm stops when the increase becomes insignificant.

The main disadvantage of supervised methods is that production of hand-labeled training corpus is expensive. In addition, the problem of sense tagging is not always well-defined (Kilgarriff, 1997), potentially causing inconsistencies and errors in the labeling process.

3.1.2 Dictionary Based Methods

In this group of methods the training corpus is unlabeled; instead external sources of knowledge such as dictionaries and thesauri are used. The advantage is that these sources are readily available and do not require any additional human effort.

Dictionary-based Disambiguation. Lesk (1986) was the first to look at word sense disambiguation as a problem for natural language processing (NLP) tasks. He describes an algorithm that syntactically parses passages of text, and then, using a machine readable dictionary, assigns a particular sense to ambiguous words. Lesk's assumption is that a word's dictionary definitions are likely to be good indicators for the sense they define. He expresses the dictionary sub-definitions of the ambiguous word as sets of bag-

of-words and the words occurring in the context of the ambiguous word as single bags-of-words emanating from its dictionary definitions (all pooled together). Disambiguation of the ambiguous word is based on choosing the sub-definition of the ambiguous word that has the greatest overlap with the words occurring in its context.

Thesaurus-Based Disambiguation. The basic idea here is that the semantic categories of the words in a context determine the semantic category of the context as a whole. This category, in turn, determines which word senses are used.

In (Walker, 87) each word is assigned one or more subject codes which correspond to its different meanings. For each subject code, we count the number of words (from the context) having the same subject code. We select the subject code corresponding to the highest count.

Yarowski(1992) describes a program that disambiguates English word senses in unrestricted text using statistical models of the major Roget's Thesaurus categories. Roget's categories serve as approximations of conceptual classes. The categories listed for a word in Roger's index tend to correspond to sense distinctions; thus selecting the most likely category provides a useful level of sense disambiguation. The selection of categories is accomplished by identifying and weighting words that are indicative of each category when seen in context, using a Bayesian theoretical framework.

Mihalcea (1999) used a WordNet based system to generate a sense tagged corpus without further supervision.

3.1.3 Unsupervised Methods: Disambiguation vs. Discrimination

Word sense disambiguation is the task of assigning sense labels to occurrences of a polysemous word. In (Schutze, 1998) this problem was divided into two sub-problems: sense discrimination and sense labeling. Sense discrimination divides the occurrences of a word into number of classes by determining for any two occurrences whether they belong to the same sense or not. Sense labeling assigns a sense to each class, and, in combination with sense discrimination, to each occurrence of the ambiguous word. Word sense discrimination can be performed without any supervision, i.e., any external knowledge of the ambiguous words meaning. The algorithm proposed in (Schutze, 1998) is based on context-group discrimination. Context group discrimination groups the occurrences of an ambiguous word into clusters, where clusters consist of contextually similar occurrences. Words, contexts, and clusters are represented in a high-dimensional, real-valued vector space. The context vectors are then clustered into coherent groups such that occurrences judged similar according to some metric are assigned to the same cluster. Clusters are represented by their centroids, the average of their elements. An occurrence in a test text is disambiguated by computing the vector representation of the relevant context, and assigning it to the cluster whose centroid is closest to that representation.

Schutze (1998) claims that "context-group discrimination can be generalized to do a discrimination task that goes beyond the notion of sense that underlies many other

contributions to the disambiguation literature. If the ambiguous word's occurrences are clustered into a large number n of clusters (e.g., $n = 10$), then the clusters can capture fine contextual distinctions. Consider the example of *space*. For a small number of clusters, only the senses of 'outer space' and 'limited extent in one, two, or three dimensions' are separated. If the word's occurrences are clustered into more clusters, then finer distinctions such as the one between 'office space' and 'exhibition space' are also discovered. Note that differences between sense entries in dictionaries are often similarly fine-grained."

Other works that use clustering based on similarity of term frequency in context include (Rosenfeld et al. 1969, Kiss 1973, Ritter et al. 1989, Pereira et al. 1993). Pederson (1997) used features for context representation and EM. Our methods is most similar to Schutze (1992 & 1998) because we will use context representation that is only based on the word co-occurrence patterns and make use of Singular Value Decomposition as part of LSA.

3.2 WSD and Psycholinguistics

Work in psycholinguistics in the 1960's and 70's established that semantic priming-- a process in which the introduction of a certain concept will influence and facilitate the processing of subsequently introduced concepts that are semantically related-- plays a role in disambiguation by humans [(see, e.g., Meyer and Schvaneveldt, 1971). This idea is realized in *spreading activation* models [(see Collins and Loftus, 1975; Anderson, 1976, 1983), where concepts in a semantic network are activated upon use, and activation spreads to connected nodes. Activation is weakened as it spreads, but certain nodes may receive activation from several sources and be progressively reinforced. McClelland and Rumelhart (1981) added to the model by introducing the notion of *inhibition* among nodes, where the activation of a node might suppress, rather than activate, certain of its neighbors. Applied to lexical disambiguation, this approach assumes that activating a node corresponding to, say, the concept MONEY will activate the "financial institution" sense of *bank*, whose activation would in turn inhibit the activation of other senses of *bank* such as "shore".

4. Latent Semantic Analysis

Latent semantic analysis (LSA) is a mathematical technique used in natural language processing for finding complex and hidden relations of meaning among words and the various contexts in which they are found [Landauer, Dumais, 1997; Landauer et al, 1998]. It is built on the basic ideas of association of elements with contexts and similarity in meaning defined by similarity in shared contexts. LSA utilizes singular value decomposition on the term-document matrix to perform the mapping between lexical space and semantic space. Although LSA is a purely a computational technique that is based only on co-occurrence patterns, it can produce results that mimic the performance of humans on certain standard language tests, including synonym-antonym matching, vocabulary and topic extraction.

This section is organized as follows: first, we will describe basic vector representation of a document known as a vector space model (VSM) and its applications. Then, we will introduce LSA and explain how it addresses some of shortcomings of VSM. In the sub-

sections that follow, we will describe the algorithm, several of its variations and its applications that are relevant to this project.

4.1 Vector Space Model (VSM)

Vector space model was introduced by Salton et al. (1975, also Salton and McGill 1983) for the field of information retrieval and had shown very good performance among the IR systems (Harman 1992). In this model, documents and queries are represented as vectors in a space where the axes (or dimensions) correspond to words. The coordinates of these vector are usually a function of term frequency (how many times term appear in a document) and/or document frequency (how many document contain this term).

Figure 1 illustrates this model. Real world examples have a vocabulary of several thousands of words but here for simplicity we assume a vocabulary of two words (i.e., "car" and "insurance") so that the lexical space has only two dimensions. 4 vectors in Figure 1 represent a query (q) and three documents (d_1, d_2, d_3) as follows:

$$d_i = \frac{(tf("car", d_i), tf("insurance", d_i))}{\sqrt{tf^2("car", d_i) + tf^2("insurance", d_i)}};$$

$$q = \frac{(tf("car", q), tf("insurance", q))}{\sqrt{tf^2("car", q) + tf^2("insurance", q)}},$$

where $tf(t, d)$ = the number of times word t occurred in document d .

For the examples shown in figure 1:

$$q = (0.71, 0.71), d_1 = (0.13, 0.99), d_2 = (0.8, 0.6), d_3 = (0.99, 0.13)$$

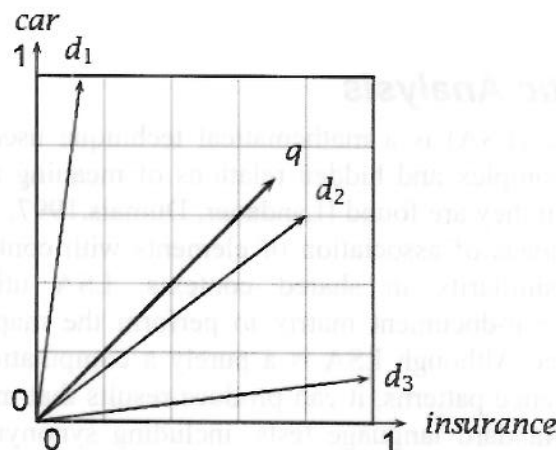
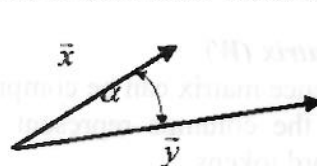


Figure 1: The document and the query vectors (Manning and Schutze, 1999)

Using VSM, we can compare documents and queries by computing a similarity measure between the vectors that represent them. The most commonly used similarity measure is

cosine similarity measure illustrated in Fig.2 which refers to the angle between two vectors (see Rhode et al. 2004 for some other measures):



$$\cos \alpha = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \|\vec{y}\|} = \frac{\sum_i x_i y_i}{\sqrt{\left[\sum_i y_i^2 \right] \left[\sum_i x_i^2 \right]}}$$

Figure 2: Cosine similarity measure (Gliozzo, 2001)

One important characteristic of this model is that the relationship between words and their sequence is lost, just like in a bag-of-words model. A known problem of this model for IR is that VSM does not take into consideration polysemy and synonymy problems.

4.2 Latent Semantic Analysis (LSA)

LSA can be looked at as an extension to VSM. It addresses the polysemy and synonymy problems by representing words and documents in a semantic space using mutual implications of words and passages. The concept of LSA was initially introduced as an indexing method called Latent Semantic Indexing (LSI) by Deerwester et al. (1990). It was later shown that this method has some similarity to how human acquire the meaning of words through the semantic similarity of word contexts and not through explicit definitions (Miller & Charles 1991, Landauer & Dumais 1997, Laham 1998, Kintsch 2002).

In the remainder of this sub-section will explain each step of LSI/LSA. For more details refer to LSI web site (<http://www.cs.utk.edu/~lsi/> M. Berry, S. Dumais) or (Rosario, 2000). <http://lsa.colorado.edu/> demos how LSA can be used for comparisons tasks (nearest neighbor, matrix comparison, etc.) All figures in this section are from Manning and Schutze (1999) (originally adapted from Deerwester et al. (1990)).

LSA can be outlined as follows:

- Choosing a context size: e.g., this can be a fix window of words, a sentence, a paragraph, a document, etc.
- Compiling a co-occurrence matrix
- Normalizing the matrix
- Singular Value Decomposition (SVD) and dimensionality reduction

4.2.1 Choosing the Context

The first decision to make when creating LSA representation is the definition of word context. In the information retrieval the context is usually the entire document (e.g., a web page), but for other applications of LSA, such as WSD, context is not uniquely defined, and various context sizes may produce different results. Schutze (1992) uses the number of characters around the word and shows that a context of 1000 character window has the best performance. Jones (1997) used the context size of 28 which is the average length of sentence in corpus used and claimed that changing window size has little effect in LSA results. Rhode et al. (2004) uses much smaller windows of size 2-7

words. We will experiments with various context boundaries such as fixed windows, sentence, paragraph and document and will compare the result with similar work.

4.2.2 Creating Document-Term Co-Occurrence Matrix (W)

After the context size has been decided a co-occurrence matrix can be compiled using the data in the corpus. In the co-occurrence matrix the columns represent the different contexts in the corpus, and the rows the different word tokens.

Figure 3 shows an example of such document-term co-occurrence matrix (document can be replaced by any context). An entry ij in the matrix corresponds to the count of the number of times the word token i appeared in context j . For example, in the context d_1 the word “cosmonaut” occurred once while in d_2 it never occurred.

	d_1	d_2	d_3	d_4	d_5	d_6
cosmonaut	1	0	1	0	0	0
astronaut	0	1	0	0	0	0
moon	1	1	0	0	0	0
car	1	0	0	1	1	0
truck	0	0	0	1	0	1

Figure 3 – Document-Term Co-occurrence Matrix W

This matrix defines a vector representation for each context. This vector representation is also called the first-order representation, in contrast with second order representation of the context (Schutze, 1998) which is using word-word co-occurrence information.

4.2.3 Normalizing the matrix

After co-occurrence matrix has been computed, it is normalized. Some weighting schemes aim at normalizing the entropy by weighting the values for each word (i.e., each set of row values) to emphasize their relative importance to the individual contexts and the set of contexts as a whole. In some cases a uniform weighting schema is used (Jones 1997, Rhode 2004). In addition, in order to prevent numerical issues in the next steps, it is common to perform a normalization of the document (or term) vectors independently into unit vectors.

4.2.4 Singular Value Decomposition (SVD) and Dimensionality Reduction

The critical step of the LSA algorithm is to compute the singular value decomposition (SVD) of the normalized co-occurrence matrix. An SVD is similar to eigenvalue decomposition, but can be computed for non-square matrices. Given any matrix W , SVD can uniquely output three matrices U , S and V such that: $W = USV^T$, where U contains orthonormal columns known as the *left singular vectors*, and V contains orthonormal rows known as the *right singular vectors*, while the middle, S , is a diagonal matrix containing the *singular values*. The left and right singular vectors are akin to eigenvectors and the singular values are akin to eigenvalues and rate the importance of the vectors.

Figures 4 through 6 shows an example of SVD for the W matrix from Figure 3.

	cosm.	astr.	moon	car	truck
Dimension 1	-0.44	-0.13	-0.48	-0.70	-0.26
Dimension 2	-0.30	-0.33	-0.51	0.35	0.65
Dimension 3	0.57	-0.59	-0.37	0.15	-0.41
Dimension 4	0.58	0.00	0.00	-0.58	0.58
Dimension 5	0.25	0.73	-0.61	0.16	-0.09

Figure 4 – Matrix U : Left Singular Vector (U is Word Vectors in Semantic Space)

2.16	0.00	0.00	0.00	0.00
0.00	1.59	0.00	0.00	0.00
0.00	0.00	1.28	0.00	0.00
0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.39

Figure 5 – Matrix S : Singular Values

	d_1	d_2	d_3	d_4	d_5	d_6
Dimension 1	-0.75	-0.28	-0.20	-0.45	-0.33	-0.12
Dimension 2	-0.29	-0.53	-0.19	0.63	0.22	0.41
Dimension 3	0.28	-0.75	0.45	-0.20	0.12	-0.33
Dimension 4	0.00	0.00	0.58	0.00	-0.58	0.58
Dimension 5	-0.53	0.29	0.63	0.19	0.41	-0.22

Figure 6 – Matrix V : Right Singular Vector (V^T is Context Vectors in Semantic Space)

The singular vectors reflect principal components, or axes of greatest variance in the data, constituting concepts of in the semantic space. If the matrices comprising the SVD are permuted such that the singular values are in decreasing order, they can be truncated to a much lower rank, s . It can be shown that the product of these reduced matrices is the best rank s approximation, in terms of sum squared error, to the original matrix W . According to (Landauer et al, 1997), it is this dimensionality reduction step, the combining of surface information into a deeper abstraction, that captures the mutual implications of words and passages, and singular vector represent the axes of this semantic space.

Figure 7 illustrated the dimensionality reduction step figure 8 shows the reduced dimensionality document vectors. The vector representing word w in the reduced rank space is U_w , the w -th row of U , while the vector representing context (document) d is V_d , the d -th row of V .

The similarity of two words or two documents in LSA is usually computed using the cosine of their reduced dimensionality vectors. Comparison of the original co-occurrence matrix W and the direction of the vectors in figure 9 demonstrates the concept of LSA.

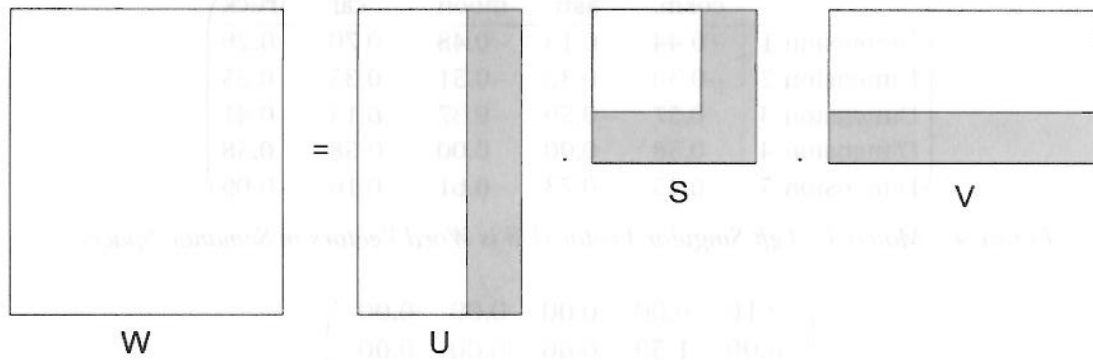


Figure 7 – Reducing Dimensionality

	d_1	d_2	d_3	d_4	d_5	d_6
Dimension 1	-1.62	-0.60	-0.04	-0.97	-0.71	-0.26
Dimension 2	-0.46	-0.84	-0.30	1.00	0.35	0.65

Figure 8 – Reduced dimensionality Matrix V

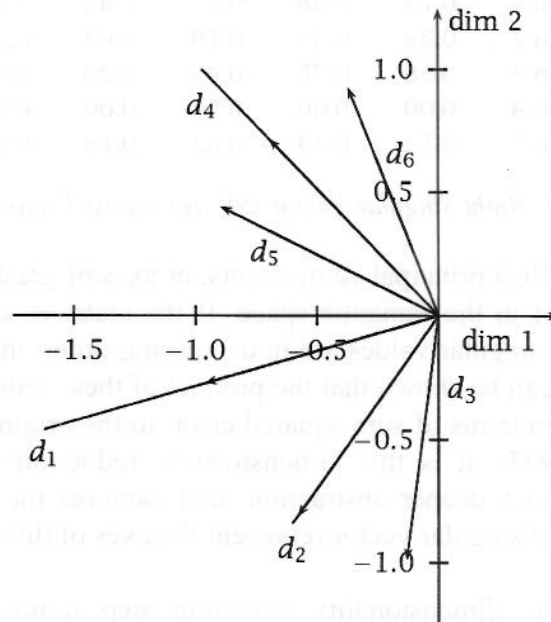


Figure 9 – Documents in Reduced Semantic Space

4.3 Extensions to LSA

4.3.1 Probabilistic Latent Semantic Analysis (PLSA)

Hofmann (1999) proposed Probabilistic Latent Semantic Analysis as a statistical technique for the analysis data co-occurrence. Compared to standard Latent Semantic Analysis which stems from linear algebra and performs a Singular Value Decomposition of co-occurrence tables, the proposed method is based on a mixture decomposition derived from a latent class model.

The starting point for Probabilistic Latent Semantic Analysis is a statistical model which has been called aspect model. The aspect model is a latent variable model for co-occurrence data which associates an unobserved class variable z with each observation. Like virtually all statistical latent variable models the aspect model introduces a conditional independence assumption, namely that d and w are independent conditioned on the state of the associated latent variable z . Since the cardinality of z is smaller than the number of documents/words in the collection, z acts as a bottleneck variable in predicting words. Hofmann (1999) argued that this approach is more principled than standard Latent Semantic Analysis, since it possesses a sound statistical foundation, and has experimentally verified the claimed advantages achieving substantial performance gains. Probabilistic Latent Semantic Analysis can thus to be considered as a promising novel unsupervised learning method with a wide range of applications in text learning and information retrieval.

4.3.2 Latent Semantic Kernel (LSK)

Cristianini et al. (2001) proposed algorithm that combines kernel methods that have successfully been used for text categorization with LSA. Kernel based learning methods are a state-of-the-art class of learning algorithms, whose best known example is Support Vector Machines (SVMs) [Burges, 1999]. In this approach, data items are mapped into high dimensional spaces, where information about their mutual positions (inner products) is used for constructing classification, regression, or clustering rules. They are modular systems, formed by a general purpose learning module (e.g. classification or clustering) and by a data specific element, called the kernel that acts as an interface between the data and the learning machine by defining the mapping into the feature space. Kernel based algorithms exploit the information encoded in the inner product between all pairs of data items. Somewhat surprisingly, this information is sufficient.

In their paper Cristianini et al. shows how LSA/LSI can be performed implicitly in any kernel induced feature space, and how it amounts to a 'kernel adaptation' or 'semantic kernel learning' step. Once the dimension of the new semantic feature space are fixed, its computation is equivalent to solving a convex optimization problem of eigenvalue decomposition, so it has just one global maximum that can be found efficiently using a technique based on the GramSchmidt orthogonalisation procedure. Experimental results are provided with text and nontext data showing that the techniques can deliver improvements on some datasets.

4.3.3 Latent Semantic Mapping (LSM)

LSM is a generalization of LSA so it can be applied to many other fields as a "mapping model" (Bellegarda, 2005). This article looks at LSA as a tool for extracting "underlying latent semantic structure that is partially obscured by randomness in choice of words". LSM is mapping data from one space to another by taking into consideration three factors (which are originated in LSA):

- 1) Discrete values are mapped to continuous parameters so they are more suitable for machine learning algorithms

- 2) Dimensionality is reduced in the direction of highest variance
- 3) Global outlook is preserved which plays an important role in abstracting large amount of data.

This model is used for applications such as junk email filtering or pronunciation modeling. The difficulty with this model is again the computational cost of SVD, especially when it comes to updating the model with a large amount of data.

5. Model-based Clustering

Clustering is the unsupervised classification of patterns (observations, data items, or feature vectors) into groups (clusters). Existing clustering approaches can be divided into two main categories: *discriminative* (distance/similarity based) approaches and *generative* (model-based) approaches. In similarity-based approaches, one determines a distance or similarity function between pairs of data samples, and then groups similar samples together into clusters. A disadvantage of the similarity based approaches is that calculating the similarities between all pairs of data samples is computationally inefficient, requiring a complexity of at least $O(N^2)$, where N is the number of samples in a dataset.

Parametric or model-based approaches, on the other hand, attempt to learn generative models from the data, with each model corresponding to one particular cluster. The type of model is often specified *a priori* (e.g. Gaussian, or Gaussian mixture models). The model structure (e.g. the number mixtures) can be selected by model selection techniques and parameters estimated using the Expectation Maximization (EM) algorithm [Dempster et al, 1977]. Probabilistic model-based clustering techniques have shown very promising results in different applications. Gaussian mixture models are the most popular models used for vector data [Banfield and Raftery, 1993; Fraley, 1999, Yeung et al, 2001]. A big advantage of the model-based clustering framework is that the similarity measure between any pair of data samples, which is usually difficult to define and calculate for certain complex data types, is not needed.

In both categories, the most popular clustering techniques include partitional clustering and hierarchical clustering [Hartigan, 1975; Jain et al, 1999]. A partitional method iteratively partitions the data samples into K (often specified *a priori*) groups according to some optimization criterion.

A hierarchical method creates a hierarchical grouping of the data samples and thus returns a set of nested clusterings.

A well-known example of model-based partitional clustering algorithms is the K-means algorithm, a simple but effective algorithm that has been used extensively in practice. Theoretically it is related to the famous EM algorithm, which is a soft partitional method when used for clustering [Alldrin et al, 2003].

The K-Means algorithm finds K clusters by choosing K data points at random as initial cluster centers. Each data point is then assigned to the cluster with center that is closest to that point. Each cluster center is then replaced by the mean of all the data points that have been assigned to that cluster. This process is iterated until no data point is reassigned to a different cluster.

EM finds clusters by determining a mixture of Gaussians that fit a given data set. Each Gaussian has an associated mean and covariance matrix. However, if spherical Gaussians are used, a variance scalar is used in place of the covariance matrix. The prior probability for each Gaussian is the fraction of points in the cluster defined by that Gaussian. These parameters can be initialized by randomly selecting means of the Gaussians, or by using the output of K-means for initial centers. The algorithm converges on a locally optimal solution by iteratively updating values for means and variances.

5.1 The use of Regularization Methods to Determine the Number of Clusters (word senses)

The problem of unsupervised clustering when the number of clusters is not known is ill-posed. The error of fitting to the data can only be reduced by increasing the number of clusters: the best fit for the data would be a trivial clustering where the number of clusters is equal to the number of vectors in our data set. For WSD that means a different sense for every occurrence of a word. Of course, this trivial clustering does not provide any information on the nature of the data. To find automatically both the best number of cluster and meaningful clustering, regularization methods can be used. Regularization is a method of imposing additional conditions for solving inverse problems with optimization methods. When model parameters are not fully constrained by the problem (the inverse problem is mathematically ill-posed), regularization limits the variability of the model and guides the iterative optimization to the desired solution by adding assumptions about the model power, smoothness, predictability, etc. A thorough mathematical theory of regularization has been introduced by works of Tikhonov's school [(Tikhonov and Arsenin, 1977)].

With regularization, the objective function that the clustering algorithms are trying to optimize measures not only the fit of the model to the data (this fit can only be made better by adding clusters), but also a penalty term that measures the complexity of the model (this penalty is smaller for simpler models with fewer clusters).

In particular, we are interested in two instances of regularization methods, namely, Minimal Description Length and Structured Risk Minimization.

5.1.1 Minimum Description Length

The Minimum Description Length (MDL) Principle [Rissanen, 1978; Rissanen, 1989; Grünwald et al, 2005] is a method for inductive inference that provides a generic solution to the model selection problem. MDL views the problem of model estimation as data compression, and is based on the following insight: any regularity in the data can be used to compress the data, i.e. to describe it using fewer symbols than the number of symbols needed to describe the data literally. The more regularities there are, the more

the data can be compressed. Equating 'learning' with 'finding regularity', we can therefore say that the more we are able to compress the data, the more we have learned about the data. Formalizing this idea leads to a general theory of inductive inference with several attractive properties:

1. Occam's Razor. MDL chooses a model that trades-off goodness-of-fit on the observed data with 'complexity' or 'richness' of the model. As such, MDL embodies a form of Occam's razor, a principle that is both intuitively appealing and informally applied throughout all the sciences.
2. No over-fitting. MDL procedures automatically and inherently protect against over-fitting and can be used to estimate both the parameters and the structure (e.g., number of clusters) of a model.
3. Bayesian interpretation. MDL is closely related to Bayesian inference, but avoids some of the interpretation difficulties of the Bayesian approach, especially in the realistic case when it is known a priori to the modeler that none of the models under consideration is true.
4. No need for 'underlying truth'. In contrast to other statistical methods, MDL procedures have a clear interpretation independent of whether or not there exists some underlying 'true' model.
5. Predictive interpretation. Because data compression is formally equivalent to a form of probabilistic prediction, MDL methods can be interpreted as searching for a model with good predictive performance on unseen data.

The Use of MDL for Clustering Word Senses.

Using MDL principle the objective function to be optimized by clustering algorithms includes two terms. The first term measures the quality of data fit – and in MDL terms, the number of bits needed to encode the data given the model, and the second term, corresponding to the penalty term in regularization techniques, measures the number of bits needed to encode the model, and is proportional to the number of parameters (clusters) in the model. An example of MDL application for automatically finding the number of clusters is AUTOCLASS (Cheeseman, 1988), a project by NASA.

5.1.2 Structural Risk Minimization

Structural risk minimization (SRM) is an inductive principle for selecting a model from a sequence of sets of models based on complexity regularization. It was introduced by Vapnik and Chervonenkis, 1974, and later analyzed by Lugosi and Zeger(1995, 1996). Structural Risk minimization is based on statistical theory of machine learning. It had been shown that the *generalization* error of model (i.e. learning machine) is bounded by a sum of two terms: the first is the error the machine makes on a training *set* of data, $R_{emp}(\alpha)$, and the second is a term that is proportional to the VC-dimension h [Vapnik and Chervonenkis, 1971] of the learning machine normalized by the size of the training set l . VC dimension of a model is usually related to the number of independent

parameters of the model. SRM is a methodology to choose a model that minimizes this upper bound on actual risk (generalization error) $R(\alpha)$

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\left(\frac{h(\log(2l/h) + 1) - \log(\eta/4)}{l} \right)}$$

over a nested structure of models as shown in figure 10.

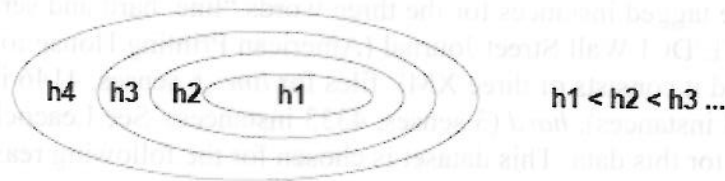


Figure 10: Structural Risk minimization (Borges (1998))

The Use of SRM for Clustering Word Senses.

To use SRM for word sense clustering, several experiments for word-sense clustering need to be performed, with varying number of clusters. The optimal clustering among those is chosen as the one that minimizes the upper bound of generalization performance.

6. Using LSA for word sense frequency estimation: Experimental Setup

Data Preprocessing:

- Stemming: Porter Stemmer (1980) will be used for extracting the roots of the words, e.g., 'eating' will be converted to 'eat'
- Eliminating stop words: Stop words are high frequency words that often do play important role for classifying the context, e.g., 'a', 'the', etc.
- Creating n-gram: n-grams are groups of n-word sequences that we will treat as words. The advantage is capturing some of the language structures in compare to bag-of-words (i.e., using unigram or treat each word separately)

Evaluation:

There are two ways to evaluate the results:

1. Perform the 'sense tagging' manually and compare the cluster memberships with tags.
2. Compute cluster purity against the most frequent tagged sense:

$$S_{i_{\max}} = \arg \max_{S_i} n(S_i)$$

$$error_i = 1 - \frac{n(x \in C_i \wedge S(x) = S_{i_{\max}})}{n(x \in C_i)}$$

Corpora

The problem of finding the word sense frequency is more similar to the word sense discrimination than word sense disambiguation (See section 3 for more details). Therefore, we only need to use the sense tags for evaluation.

Senseval project started in 1998 with the goal of creating a unified standard for evaluating the word sense disambiguation systems and it provides a great source of tagged and untagged corpora, which are publicly available (See <http://www.senseval.org/> for detail). Senseval training task that is relevant to this project is called English Lexical Sample. While we will still use data provided for this task, but for our initial experiment we decided to use tagged instances for the three words “line, hard and serve”. The source of this data is ACL/DCI Wall Street Journal (American Printing House for the Blind, San Jose Mercury) and it consists of three XML files for *line* (6 senses, 4146 instances), *serve* (4 meaning, 4378 instances), *hard* (3 senses, 4333 instances). See Leacock et al. 1993 for more description for this data. This dataset is chosen for the following reasons:

1. Easy and distinguishable meaning
2. One meaning per instance
3. Existence of benchmarking results (Sensecluster by Pederson)
4. Large number of instances

In addition to that, only for training, we will use the some texts from Project Gutenberg, which is likely to improve the accuracy of the results.

Software

SVD step is perhaps the most significant and computationally speaking, the most expensive step for LSA (Asymptotically, the generic algorithm for a dense matrix is $O(n^3)$ where n is the dimension). The choice of programming language is mostly affected by this and we have decided to use the combination of the following for different stages of our experiment:

1. **Matlab** is commercial mathematical software by Mathworks, which has great feature for matrix manipulation, statistics and visualization. It has become standard de facto in research community due to its ease of use. Although various packages are available for dealing with SVD (commands: svd, ssvd and gsvd), but Matlab is usually having problems with large matrices (i.e., several thousand dimension). Also, no special consideration is made for sparse matrices however the program can be extended to support those algorithms.
2. **Java**. Very versatile programming language and many useful packages are available. JAMA is software package provided by NIST that we will use for matrix manipulation and SVD. The programs are tested to run in reasonable amount of time for up to 3000 dimensions.
3. **C++/C**. There aren't as many great ready-to-use features as in Matlab or Java, but speed and memory usage is much more in control of the programmer. Most important advantage is the availability of the SVDPACK (Berry, 1992), which computes the SVD

for sparse matrices very efficiently and has been used in most of the cited papers for this project.

References

1. N. Alldrin, A. Smith, D. Turnbull, "Clustering with EM and K-Means", University of San Diego, California, Tech Report (2003)
2. J. R. Anderson, J. Robert, *Language, Memory, and Thought*. Lawrence Erlbaum and Associates, Hillsdale, New Jersey (1976)
3. J. R. Anderson, J. Robert, A Spreading Activation Theory of Memory, *Journal of Verbal Learning and Verbal Behavior*, **22**(3), 261-95 (1983)
4. J. R. Anderson, C. Lebiere, *The Atomic Components of Thought*. Mahway, NJ: LEA (1998)
5. J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin. An Integrated Theory of the Mind. *Psychological Review* (in press)
6. AUTOCLASS <http://ic.arc.nasa.gov/ic/projects/bayes-group/autoclass/>
7. J. T. Ball, Double R Grammar (2003) <http://www.DoubleRTheory.com/DoubleRGrammar.pdf>
8. J. T. Ball A Cognitively Plausible Model for Language Comprehension, *Proc. of 13th Conference of Behavior Representation in Modeling and Simulation* (2004)
9. J. D. Banfield and A. E. Raftery, "Model-based gaussian and non-gaussian clustering," *Biometrics*, vol. 49, pp. 803-821 (1993)
10. J. Barwise, J. R. Perry, *Situations and attitudes*, MIT Press, Cambridge, Massachusetts (1983)
11. J. R. Bellegarda, Latent Semantic Mapping, *IEEE Signal Processing Magazine*, vol. 22, pp.70-80 (2005)
12. M. W. Berry, Large scale singular value computations, *International Journal of Supercomputer Applications*, 6(1):13-49, 104 (1992)
13. P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, R. L. Mercer, Word-Sense Disambiguation Using Statistical Methods, *Meeting of the Association for Computational Linguistics* (1991)
14. C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, *Data Mining and Knowledge Discovery* (1998)
15. P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, & D. Freeman, AutoClass: A Bayesian Classification System, *Proceedings of the Fifth International Conference on Machine Learning*, pp. 54-64 (1988)
16. A. M. Collins, E. F. Loftus, "A spreading activation theory of semantic processing." *Psychological Review*, **82**(6), 407-428 (1975)
17. N. Cristianini, Latent Semantic Kernels, *Proceedings of ICML-01, 18th International Conference on Machine Learning*, (2001)
18. S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, R. Harshman, Indexing by Latent Semantic Analysis, *Journal of the American Society of Information Science* (1990)
19. Dempster, N. Laird, and D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, vol. 39, no. 1, pp. 1-38 (1977)

20. Chris Fraley, "Algorithms for model-based Gaussian hierarchical clustering," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 270–281 (1999)
21. W. A. Gale, K. W. Church, D. Yarowsky, A method for disambiguating word senses in a large corpus, *Computers and the Humanities*, 26(5):415-439 (1993)
22. Gliozzo, E. D'Avanzo, C. Strapparava, Automatic Acquisition of Domain Information for Lexical Concepts, *Proc. of the Workshop Meaning*, pp. 75-80 (2005)
23. Project Gutenberg <http://www.gutenberg.org>
24. P. Grünwald, A Tutorial introduction to the minimum description length principle, *Advances in Minimum Description Length: Theory and Applications*, MIT Press (2005)
25. John A. Hartigan, *Clustering Algorithms*, John Wiley & Sons (1975)
26. D. Harman, Overview of the first text retrieval conference (TREC-1). In *Proceedings of the First Text Retrieval Conference (TREC-1)*, pp 1-20 (1992)
27. T. Hofmann, Probabilistic Latent Semantic Analysis, *Proc. Uncertainty in Artificial Intelligence* (1999)
28. N. Ide, J. Véronis, Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1), 1-40 (1998).
29. Anil K. Jain, M. Narasimha Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323 (1999)
30. JAMA: Java Matrix Package <http://math.nist.gov/javanumerics/jama/>
31. M. P. Jones, J. H. Martin, Contextual Spelling Correction Using Latent Semantic Analysis, *Proc. 5th Conference on Applied Natural Language Processing* (1997)
- Kilgarriff, "I don't believe in word senses" , *Computers and the Humanities* 31(2), pp91–113 (1997)
32. W. Kintsch, On the notions of theme and topic in psychological process models of text comprehension, *Thematics : Interdisciplinary Studies*, 157-170 (2002)
33. G. R. Kiss. Grammatical Word Classes: A Learning Process and its Simulation, *Psychology of Learning and Motivation* 7. ppl-41 (1973)
34. D. Laham, Cluster Analysis of LSA Similarities for Word-Sense Disambiguation in Context (1998)
35. T. K. Landauer, S. T. Dumais, A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge, *Psychological Review*, 104(2):211-240 (1997)
36. C. Leacock, M. Chodorow, G. A. Miller, Using Corpus Statistics and WordNet Relations for Sense Identification. In *Computational Linguistics*, volume 24, pp. 147-165 (1998)
37. C. Leacock, G. Towell, E. Voorhees, Corpus-Based Statistical Sense Resolution, *Proc. ARPA Human Language Technology Workshop* (1993)
38. M. Lesk, Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone, *Proceedings of the 5th Annual International Conference on Systems Documentation*, pp 24-26 (1986)
39. "Line, hard, serve" Data <http://www.d.umn.edu/~tpederse/data.html>
40. Gábor Lugosi and Kenneth Zeger, "Nonparametric Estimation via Empirical Risk Minimization," *IEEE Transactions on Information Theory*, **41**(3), 677–687, (1995)
41. Gábor Lugosi and Kenneth Zeger, "Concept Learning Using Complexity Regularization," *IEEE Transactions on Information Theory*, **42**, 48–54, (1996)

42. C.D. Manning, H. Schutze, Foundations of statistical natural language processing, *MIT Press* (1999)
43. Matlab <http://www.mathworks.com/>
44. MDL Research Website <http://www.mdl-research.org/>
45. J. L. McClelland, D. E. Rumelhart, "An interactive activation of context effects in letter perception: part 1. An account of basic findings." *Psychological review*, 88, 375-407 (1981)
46. D. E. Meyer, R. W. Schvaneveldt, "Facilitation in recognizing pairs of words: Evidence of a dependence between retrieval operations." *Journal of Experimental Psychology*, 90(2), 227-34 (1971)
47. R. Mihalcea, D. Moldovan, An Automatic Method for Generating Sense Tagged Corpora, *AAAI/IAAI* (1999)
48. G. A. Miller, WordNet: A lexical database for English, *CACM*, 38(11):39-41 (1995)
49. G. A. Miller, W.G. Charles, Contextual correlates of semantic similarity, *Language and Cognitive Processes*, 6, 1-28 (1991)
50. R. Mooney, Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning, *Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing (EMNLP'96)*, pp. 82-91 (1996)
51. H.T. Ng, Exemplar-Based Word Sense Disambiguation: Some Recent Improvements. In C. Cardie and R. Weischedel, eds., *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 208-213. Association for Computational Linguistics, Somerset, New Jersey (1997)
52. M. F. Porter, An algorithm for suffix stripping (1980)
53. T. Pedersen, R. Bruce, Distinguishing Word Senses in Untagged Text, *EMNLP 2*, pp.197-207 (1997)
54. F. Pereira, N. Tishby, L. Lee, Distributional Clustering Of English Words, *ACL 31*, pp.183-190 (1993)
55. J. Rissanen, Modeling by the shortest data description. *Automatica* 14, 465-471 (1978)
56. J. Rissanen, Stochastic Complexity in Statistical Inquiry. World Scientific Publishing Company (1989)
57. H. Ritter, T. Kohonen, Self-organizing semantic maps, *Biological Cybernetics*, 62:241-254 (1989)
58. D. L. T. Rodhe, L. M. Gonnerman, D. C. Plaut, An Improved Method for Deriving Word Meaning from Lexical Co-Occurance (2004)
59. B. Rosario, Latent semantic indexing: An overview, *Technical Report INFOSYS 240 Spring Paper, University of California, Berkeley* (2000)
60. Rosenfeld, H. Huang, and V. Schneider, An application of cluster detection to text and picture processing, *IEEE Transactions on Information Theory*, 15(6) (1969)
61. G. Salton, A. Wong, C. S. Yang, A vector space model for information retrieval, *Communications of the ACM*, 18(11):613-620 (1975)
62. G. Salton, M. McGill, Introduction to Modern Information Retrieval, *New York - McGraw Hill* (1983)
63. H. Schütze, Dimensions of Meaning, *Proc. Supercomputing*, IEEE Computer Society Press, Los Alamitos, California. 787- 796 (1992)

64. Schütze, Hinrich "Word space." In Hanson, Stephen J.; Cowan, Jack D.; and Giles, C. Lee (Eds.) *Advances in Neural Information Processing Systems 5*, Morgan Kauffman, San Mateo, California, 5, 895-902 (1993)
65. H. Schütze, J. O. Pedersen, A co occurrence-based thesaurus and two applications to information retrieval, *Information Processing and Management*, 33(3), p. 307-318 (1997)
66. H. Schütze, C. Silverstein, Projections for Efficient Document Clustering, *Proc. 20th International ACM SIGIR Conference* (1997)
67. H. Schütze, Automatic Word Sense Discrimination, *Journal of Computational Linguistics*, Volume 24, Number 2 (1998)
68. Senseval Project <http://www.senseval.org/>
69. SVDPACK <http://www.netlib.org/svdpack/>
70. SVM Website <http://www.svms.org/>
71. Tikhonov, A. N., and Arsenin, V. Y., 1977, Solution of ill-posed problems: John Wiley and Sons.
72. Vladimir N. Vapnik and Aleksei Ja. Chervonenkis, "Ordered Risk Minimization (I and II)", *Automation and Remote Control*, **34**, 1226–1235 and 1403–1412 (1974).
73. Vladimir N. Vapnik and Aleksei Ja. Chervonenkis, "On the Uniform Convergence of Relative Frequencies of Events to their Probabilities," *Theory of Probability and Applications*, **16**, 264–280 (1971)
74. V. Vapnik, Estimation of Dependencies Based on Empirical Data, 1979 [in Russian - English translation, Springer Verlag 1982]
75. D. Walker, Knowledge Resource Tools for Accessing Large Text Files, in *Machine Translation: Theoretical and Methodological Issues*, Serges Nirenberg, ed., Cambridge University Press, Cambridge, England (1987)
76. Y. Wilks, M. Stevenson, *The grammar of sense: Is word sense tagging much more than part-of-speech tagging?*. Technical Report CS-96-05, University of Sheffield, Sheffield, United Kingdom (1996)
77. D. Yarowsky, Word-Sense Disambiguation Using Statistical Models of Roget's Categories Train on Large Corpora, *Proceedings of COLING-92* (1992)
78. K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo, "Model-based clustering and data transformations for gene expression data," *Bioinformatics*, vol. 17, no. 10, pp. 977–987 (2001)